Project Design and Management 300
FreedomSpace

To the Moon

Delan Azabani
Callum Boyd
Jason Giancono
Robert Lai
Luke Mercuri
Jasmine Quek
Kye Russell

June 1, 2014

Contents

List of Figures

1 (TODO: Jason) Abstract

2 Introduction and objectives

The use of computers and information technology has increasingly become an integral factor in research at universities. While the technology and its role has increased, Curtin University still uses traditional methods such as paper forms for allocating storage for research projects. In order for us to keep up with the increasing demand of information technology services, many of these methods should be digitised and automated.

In order to eliminate this needless paperwork and bureaucracy, Curtin University could implement a web-based portal for staff members to request and allocate storage space for each project. This would reduce time loss in the turnaround caused by waiting for requests, improve the efficacy of record maintenance and offer extra opportunities for Curtin University to analyse and optimise its data storage infrastructure.

2.1 Objectives

The objective of this project is to create a prototype of a web-based portal for managing storage space requests at Curtin University. This will allow us to demonstrate the concept to the stakeholders and refine it as needed. The prototype isn't required to have the complete functionality, however the user experience should be complete. The prototype should focus on user experience, with usability being the main measure of success, as many staff members may not be proficient with the use of technology.

2.2 Scope

The scope of the project will include gathering the user requirements of the system, designing the user experience of the system and implementing a prototype of this user experience. Actual functionality and requirements regarding integration with Curtin University's information systems is outside the scope of this project, and will depend on the reception of the prototype by Curtin University's information technology department.

2.3 Limitations

The project has not been provided with a budget; all software dependencies must be freely available, and all hardware must be sourced from group members and/or computing resources provided by Curtin University.

The prototype will be developed on academic versions of the Visual Studio 2013 Professional integrated development environment, which means that no part of this prototype may be deployed commercially.

The developers will not have access to Curtin University's LDAP directory service and other information systems. This means that the prototype's main focus will be on the user experience and the backend which facilitates that. Enhancements such as integration with learning management systems and directory services will be features likely available in a final product but will not be represented in the prototype.

There are only twelve weeks to complete the entire project, including the solicitation of requirements and user testing. Developers will also provide different sets of skills, while being restricted by different unit timetables.

2.4 Approach

Our team is developing the prototype using the Scrum methodology. First the Product Backlog is decided upon — each item is allocated a priority, then a subset of the user stories are chosen to be completely designed, implemented and tested for each sprint. The team meets before and after each sprint to discuss how the sprint went, as well as what needs to be completed in the next sprint. This allows us to efficiently adapt to any issues with our requirements.

2.5 Structure

This report will guide us through the background which has led to this project, the requirements which have been decided upon for the prototype, the design of the web-based portal and the process used during development of this project.

3 Background

3.1 Other systems considered

Since 2013, for large volumes of researchers' storage, Curtin University has used iVEC to procure its data storage. iVEC is a computing and data storage centre whose headquarters are based in Technology Park right next to Curtin University. Because they service all four of Western Australia's major universities, there is no integration with Curtin University's information technology systems. This makes applying for storage a tedious and bureaucratic process.

The iVEC FAQ (iVEC 2013) states (emphasis by the author):

> For individuals who are granted an allocation on Data Stores @ iVEC, **a prerequisite is an iVEC account**. All researchers using iVEC Data Stores should contact help@ivec.org to have an account provided.

Intervention by iVEC is required in many portions of the storage space procurement process and this causes unnecessary delays. The procurement process also requires researchers to repeat a significant amount of information which would, in a system optimised for Curtin University, could be automatically generated using its LDAP directory service and other information systems.

3.2 Initial project management approach

Our solution approach was to first prioritise the user stories given, so that vital functionality would be implemented in early sprints. Shortly afterwards, the team had a meeting to discuss the various permissions of the different user account roles. In this meeting we made a permissions table, as shown in Figure 1, so as to clear up any confusion surrounding the wide variety of user privileges.

The overarching method was to figure out the solutions for the larger problems as a group, and then to allocate tasks to be solved by assorted group members by the end of the next sprint. Existing free technologies were used to coordinate project management and communications, including a Facebook group, Google form for time logging, Trello to organise user stories as well as Basecamp. The time allocated to various tasks in the project can be seen in the Gantt chart in Figure 2.

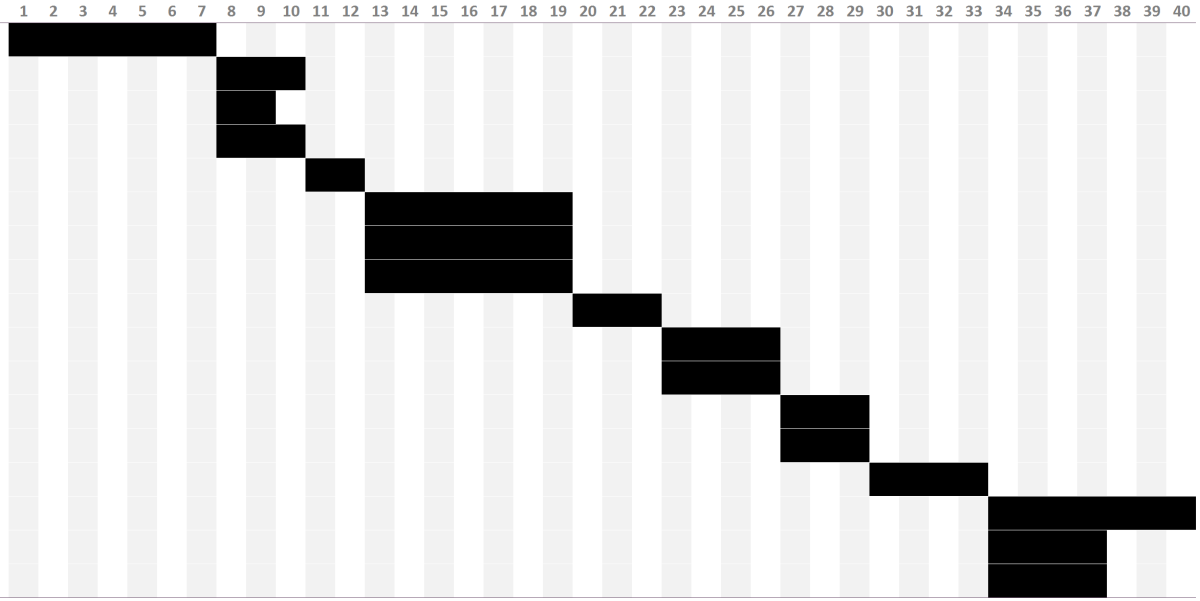
Figure 1: The user access permissions model for this prototype

```
                Receive approval notifications
                | View available spaces
                | | Request access to space
                | | | Grant/revoke access to space
                | | | | Promote/demote Data Manager
                | | | | | Request additional space
                | | | | | | Approve new/expanded space
                | | | | | | | Set Pri.Inv. owning space
                | | | | | | | | Comment on requests
                | | | | | | | | |
       Admins   X X X X X X X X X
    Approvers   X X X X X X X X
Data Managers   X X X X X X
  Researchers   X X X
```

Figure 2: The Gantt chart for this project

Project Plan

ACTIVITY	PLAN PRIORITY	PLAN START	PLAN DURATION
Global login	1	1	7
View storage spaces	1	8	3
Submit storage request (PI)	1	8	2
Review storage requests (Approver)	1	8	3
Approve storage requests (Approver, Admin)	1	11	2
User access level login	2	13	7
Granting storage access (Data Manager)	2	13	7
Grant researchers manager-level access (PI)	2	13	7
Additional storage request (Data Manager)	2	20	3
Change PI of a storage space (Admin)	2	23	4
View a list of all storage requests (Approver)	2	23	4
Send appropriate notifications (Admin)	3	27	3
Receiving email notifications (DM, PI, Researcher)	3	27	3
Viewing status of requests (DM, PI)	3	30	4
Revoking access to storage spaces (DM, PI)	3	34	7
Add comment to a request (Admin)	3	34	4
View a list of all storage spaces for a faculty (Approver)	3	34	4



3.3 Tools considered for development

- Django is a 'high-level Python Web framework' that adheres to the model-view-controller pattern. (Django Software Foundation 2014)
- XAMPP is an 'Apache distribution containing MySQL, PHP, and Perl' with a unified installation and configuration workflow for Windows, Mac OS X and Linux. (Apache Friends 2014)
- Flask is a 'microframework for Python' which provides template rendering and request routing, but avoids enforcing any particular patterns. (Ronacher 2014)
- Bootstrap is a frontend framework with HTML, CSS and optionally JavaScript-based 'reusable components'. (Twitter, Inc. 2014)
- ASP.NET MVC 5 is a 'framework for building scalable, standards-based web applications using well-established design patterns and the power of ASP.NET and the .NET Framework.' (Microsoft Corporation 2014)
- Basecamp is a project management tool that provides scheduling, task management, milestones and defect tracking. (37signals, LLC. 2004)
- Facebook Groups allow text posts, images, files and questions to be shared privately among a set of users. (Facebook, Inc. 2014)
- Trello is a web-based group project manager that implements the *kanban* paradigm. (Fog Creek Software, Inc. 2014)
- draw.io is a 'free online diagram drawing application' which can create UML, entity-relationship, network and many other types of diagrams. (JGraph Limited 2014)
- Google Drive consists of cloud storage, live collaborative document editing, and a tool for online forms. (Google Inc. 2014)
- GitHub provides 'powerful collaboration, code review, and code management for open source and private projects.' (GitHub, Inc. 2014)

3.4 Rationale for using our tools

The tools we decided to use throughout the life of the project are:

- Backend: ASP.NET MVC 5;
- Frontend: Bootstrap;
- Project management: Facebook Groups, Basecamp and Trello;
- Version control: GitHub;
- Diagrams: draw.io; and
- Documentation: Google Docs and LaTeX.

The group encountered difficulty deciding on what framework we would use for the project, as few of us had much exposure or experience in backend web development. We decided that it would be voted upon, and as a subset of the group were all familiar with C#, we swayed towards ASP.NET MVC 5.

As ASP.NET MVC 5 comes packaged with Bootstrap, it was consequently common sense to use both alongside each other for the project. The team agreed that Bootstrap was most suitable, and as a further plus it had stock templates that would help us in early development stages.

Shortly after a group was formed, as we were all Facebook friends with each other it seemed almost natural to start a Facebook Group for easy communications outside of tutorials and seminars. The ability to design polls for members of the group was a huge plus as it made it a lot easier to organise meetings, and especially so for our relatively large group of seven people.

A Basecamp project was also set up for the group assignment. We originally chose to use Basecamp because of its various services such as to-do lists and file sharing. However, we did not get much of a chance to use it as we did most task allocation in person during Scrum and team meetings. Additionally, many of the addons that were desirable for the documentation side of the project were rather costly and deterred us from using it.

Our Trello board was only set up after the group had task allocation issues as can be seen in the review of the first sprint. A few members of the group didn't originally want to use Trello as a project management tool due to seeing it used to a messy result elsewhere. However after group members were mistakenly working on the same tasks it was decided it was necessary. We used one board for the entire project, with user stories as cards, and several lists depicting the stages of the completeness of those user stories.

Alas, by this point we were using too many project management tools and it was inconvenient for people to regularly check on several applications, websites and devices. The group then ended up having the minute taker post all meeting minutes onto the Facebook Group for easy access. Trello ceased to be used after the second sprint.

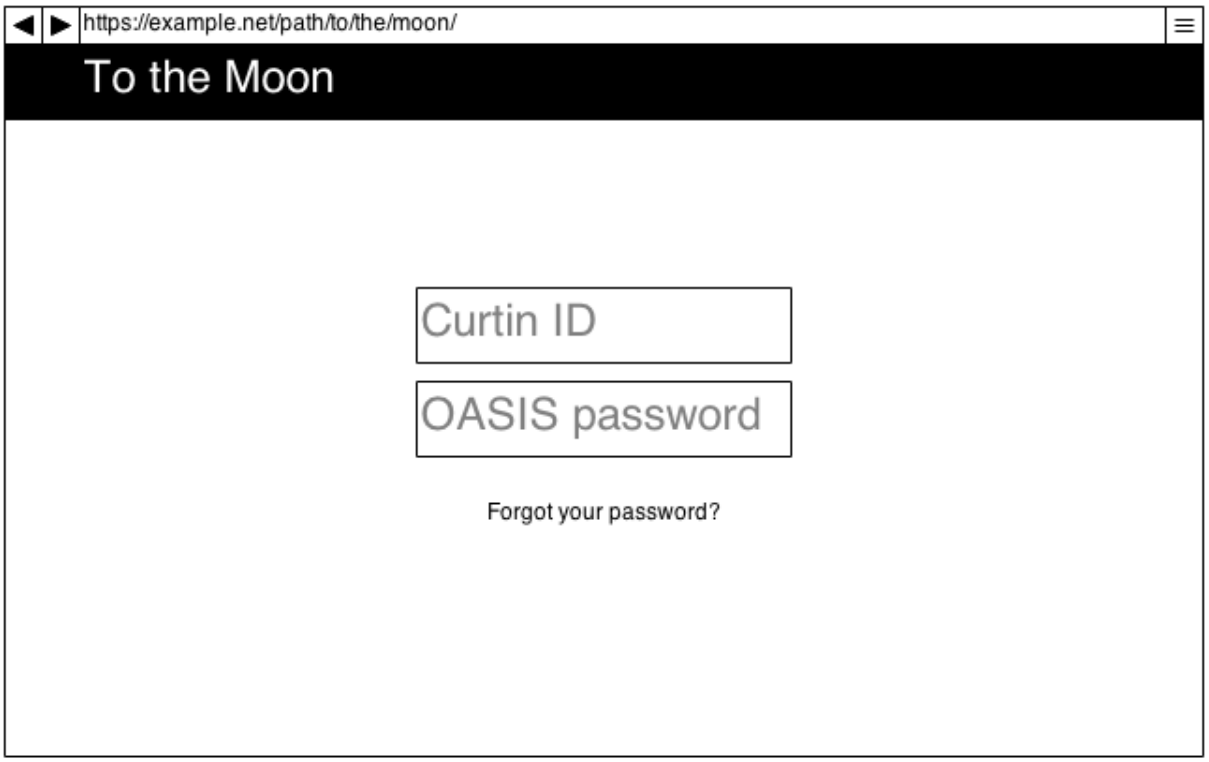
As several group members were familiar with GitHub and were very pleased with its offerings, it was suggested to be used in the project. GitHub also allows for several private repositories if signed up with an educational email address which was another benefit, as we didn't want to risk other groups plagiarising our hard work.

draw.io was used before the unofficial 'sprint zero' presentation to display our initial mockups of our website. At this point we had not started any development, and the focus was on planning the first sprint. The group decided to use Draw.io as a couple of members had used it previously for another group assignment where various UML diagrams and mockups were drawn. The only downside is that it does not support online live collaboration, but that wasn't too much of an issue. This tool was also then used for the final UML and ER diagrams for the project as documented in this report.

Google Docs was used heavily in the creation of the final report, as well as communicating between members progress of web backend functionality. A group member suggested using Google Docs and this was found in agreement with other members. One of its greatest merits is its ability to support online live collaboration. When working among a group of seven busy university students, this support is necessary. This feature was extremely helpful in seeing where other group members were up to in their allocated section of the report, as well as being able to refer to others' sections when working on one's own sections.

The final report was transcribed from a Google Docs file to be typeset using LaTeX, in order to exercise a greater degree of control over the aesthetics and typography, while automating tedious tasks such as figure layout and bibliography management.

Figure 3: A mockup for the login screen



3.5 User interface mockups

A handful of user interface mockups were created for the initial project presentation. These were completed before any development of the prototype had started, and can be seen in Figures 3, 4, 5, 6 and 7.

Figure 4: A mockup for the dashboard as seen by an administrator

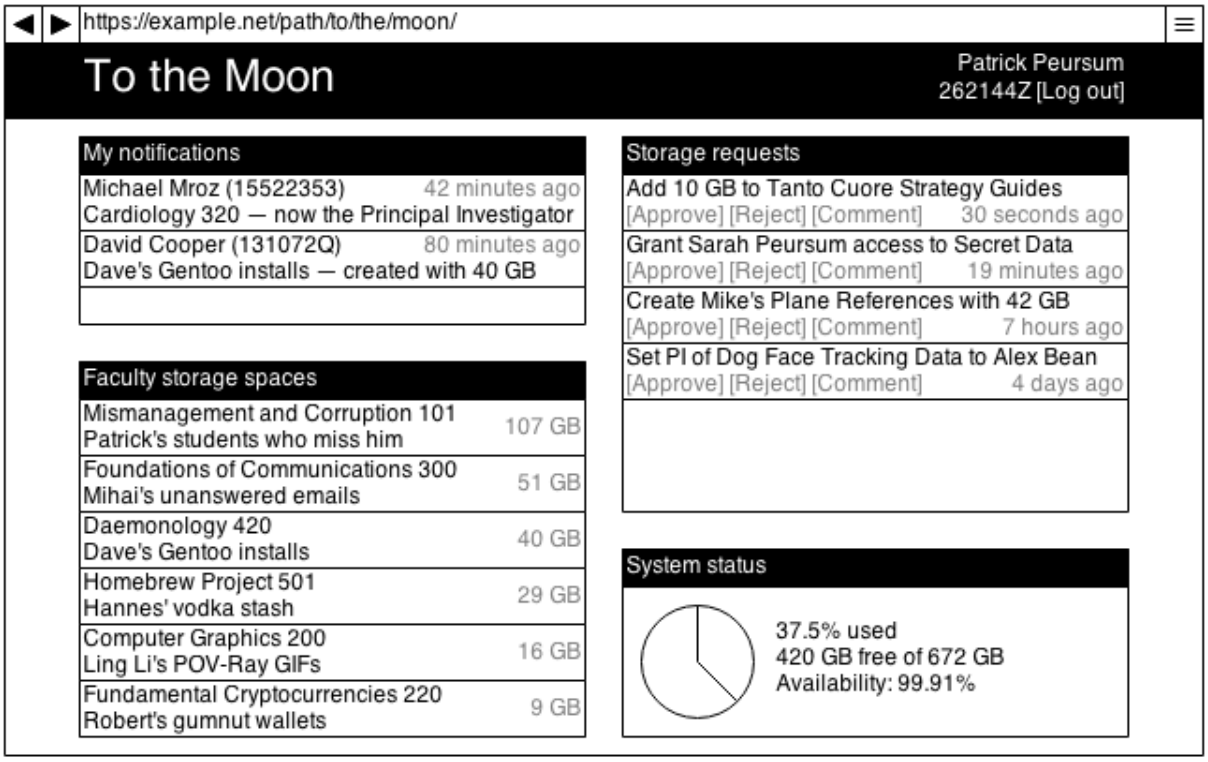


Figure 5: A mockup for the dashboard as seen by a researcher

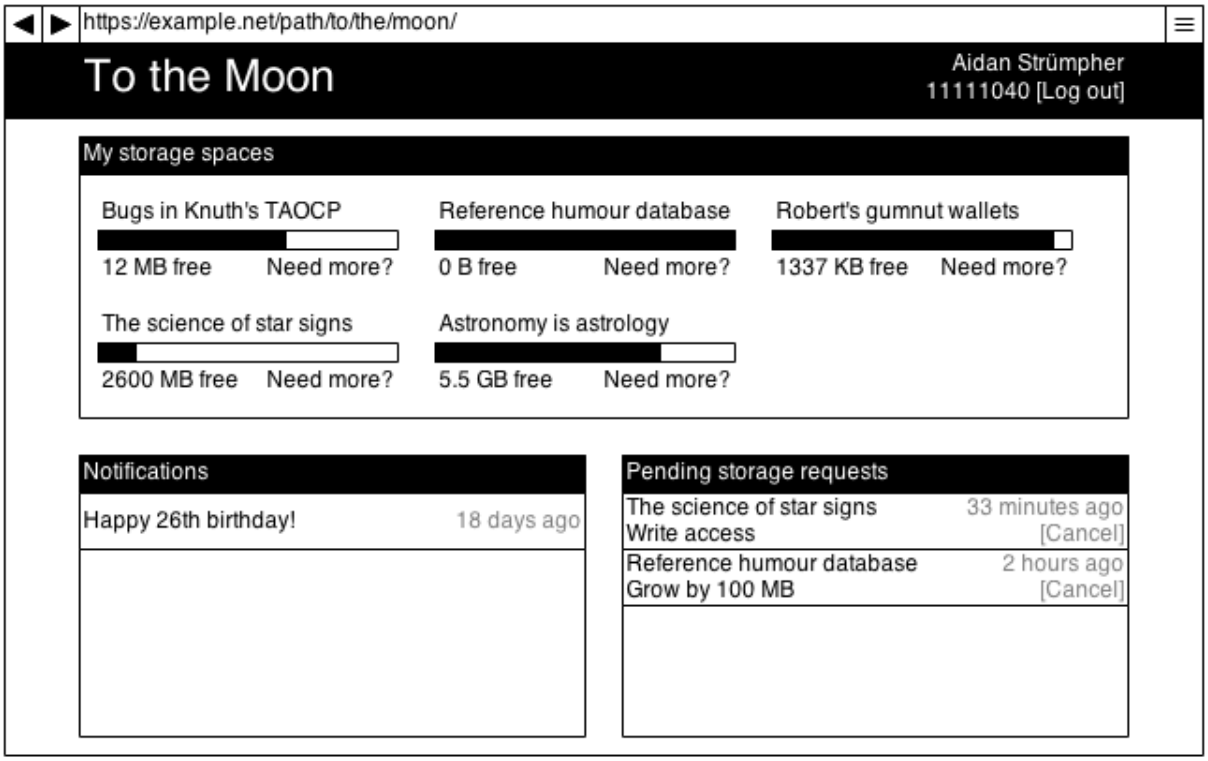


Figure 6: A mockup for the preferences screen

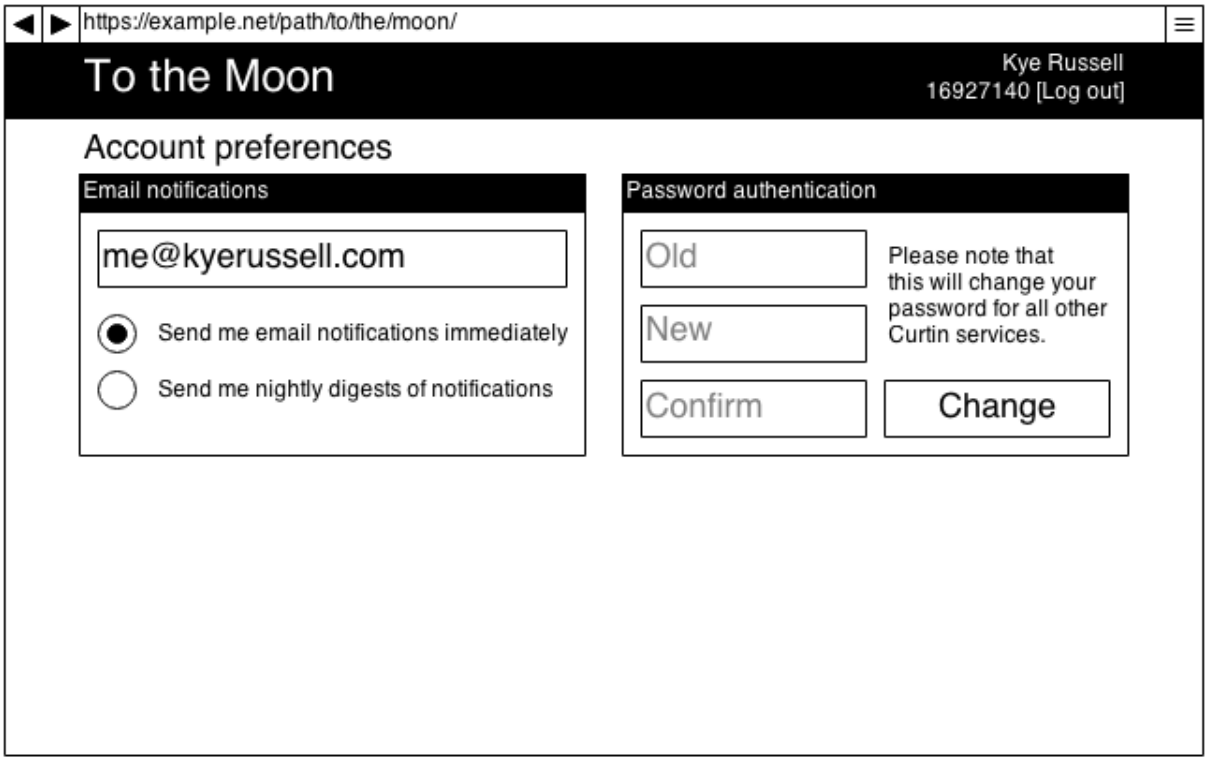
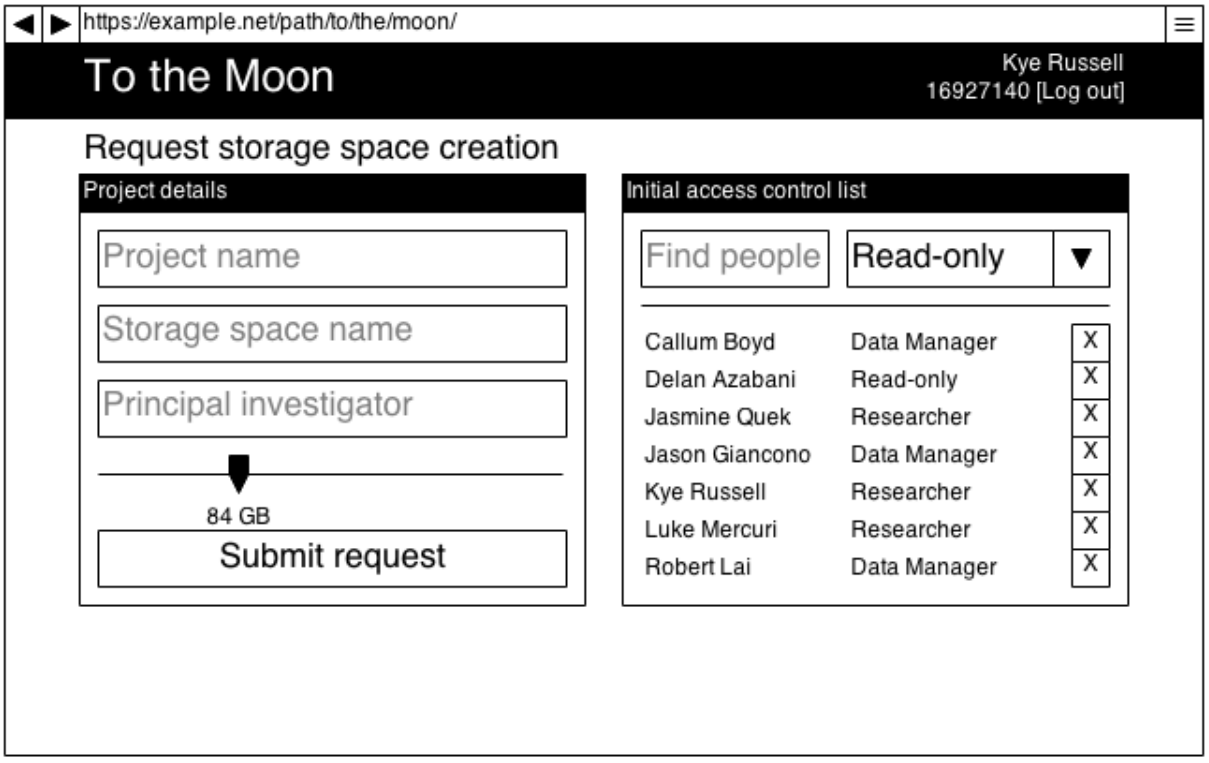


Figure 7: A mockup for the storage space request form



4 Product backlog

Our first goal with user stories was to divide them into three groups representing their priority for completion, with '1' being the highest and '3' being the lowest. In the highest priority group we placed stories that we believed to be related to core parts of the system. The second group were stories that enhanced the core functionality of the system and added features and other useful tools for the users. The lowest priority group were stories that enhanced ease of access and usability as well as features that were not core functionality.

In the first priority tier of user stories, we placed any stories relating to authentication and user access levels. We felt that this was core to the system because the users would have different permissions and access. There were quite a few stories relating to logging in and being able to perform their functions from that. Out of the seven user stories that were decided to be 'priority one', three were relating to login functions.

- As an administrator, I want to login so that I can review and approve requests;
- As a user, I want to login so that I can perform appropriate functions applicable to my role; and
- As an approver, I want to login so that I can approve storage requests.

In the first story we understand that the administrators of the system want to be able to log in and approve requests. From this we can assume that they want to be able to log in so that they will be able to have access to this feature, not just anyone.

We can further assume that this is also related to wanting some level of system and user security so that only authorised personnel have access to requests. This is also seen in the third story where the approver wants to be able to log in to approve requests. From this we can see that approving requests is a privileged feature and only these two types of users are able to do this. This can be seen in the login screen feature shown in Figure 8.

The second user story shows that all users do want to be able to log in and perform functions applicable to their role. This can be expanded to users want to be able to log in and only see functions related to their own role, therefore only displaying functions that they can use. From this assumption we can work out that users only want to see what they can do and they don't want to see everything else, as illustrated in Figure 9 and Figure 10.

While this may seem like an ease of use story it is important to make sure that all users can log in as soon as the first part of the working system is complete; therefore this was classified as a first priority story.

- As a principal investigator, I want to submit a storage request so that I can have access to storage space for a research project;
- As a user, I want to view a list of my storage spaces so that I know my access level for my research projects;
- As an administrator, I want to approve a request so that storage can be provisioned, expanded or access permissions changed; and

Figure 8: The login screen

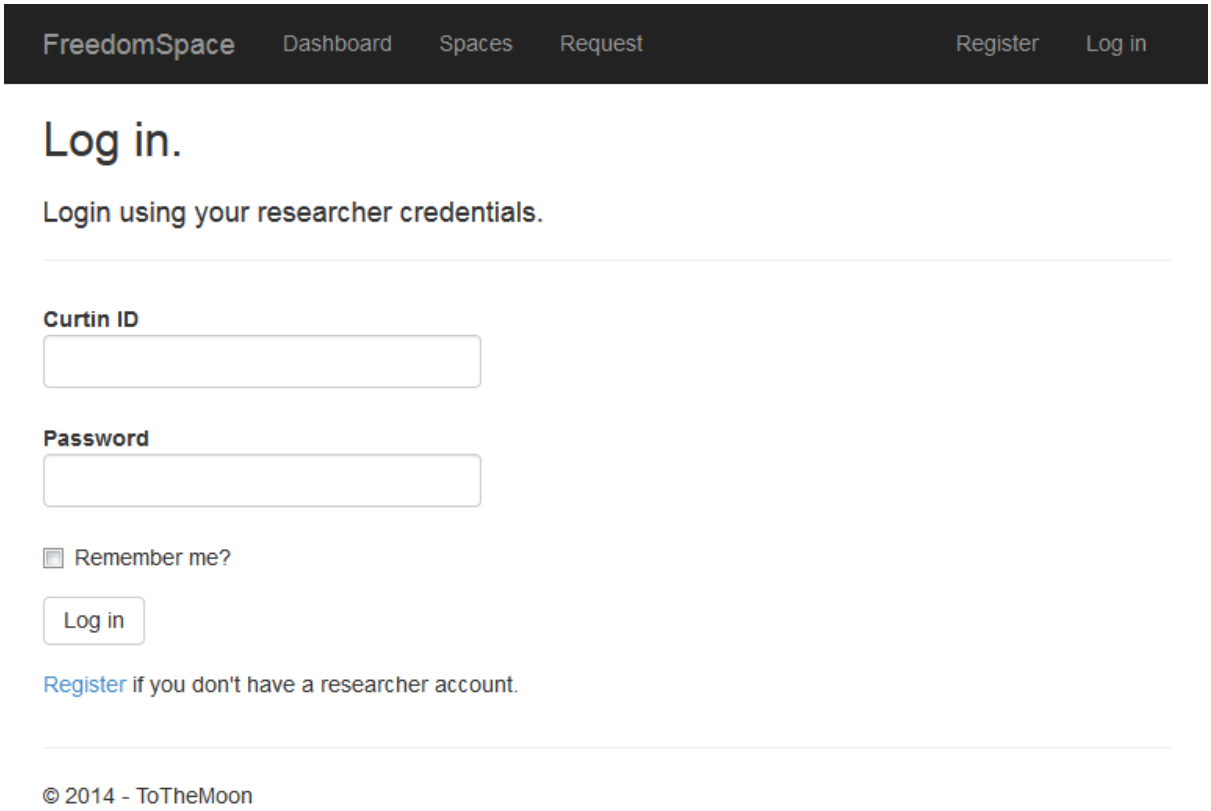


Figure 9: The dashboard as seen by a plebeian user

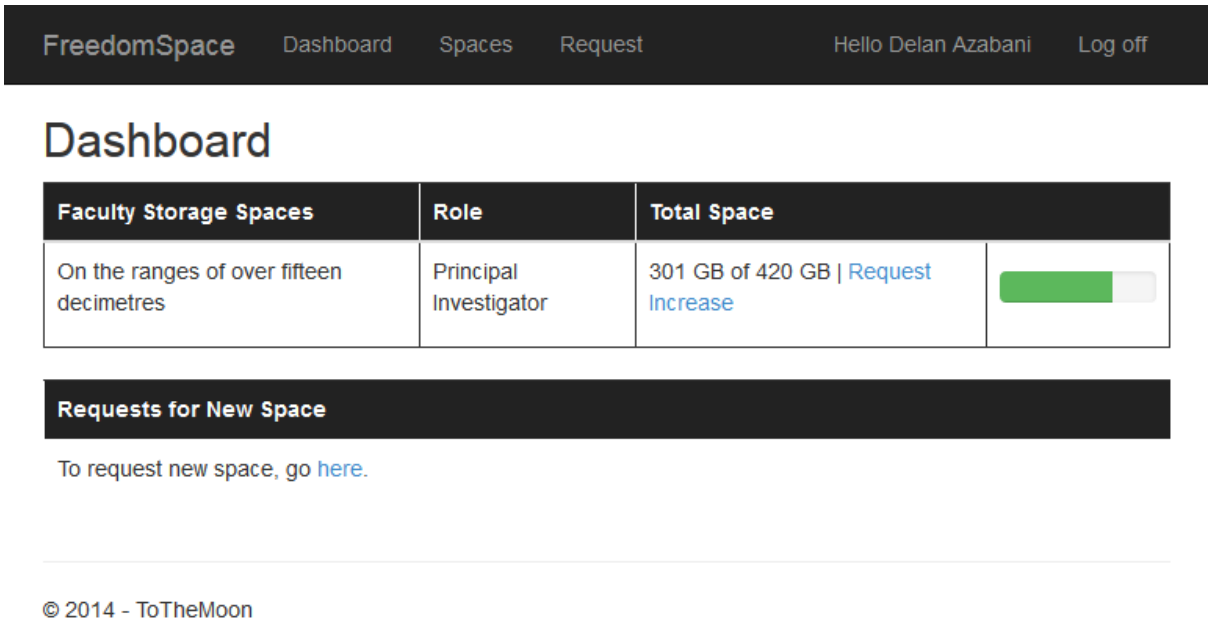
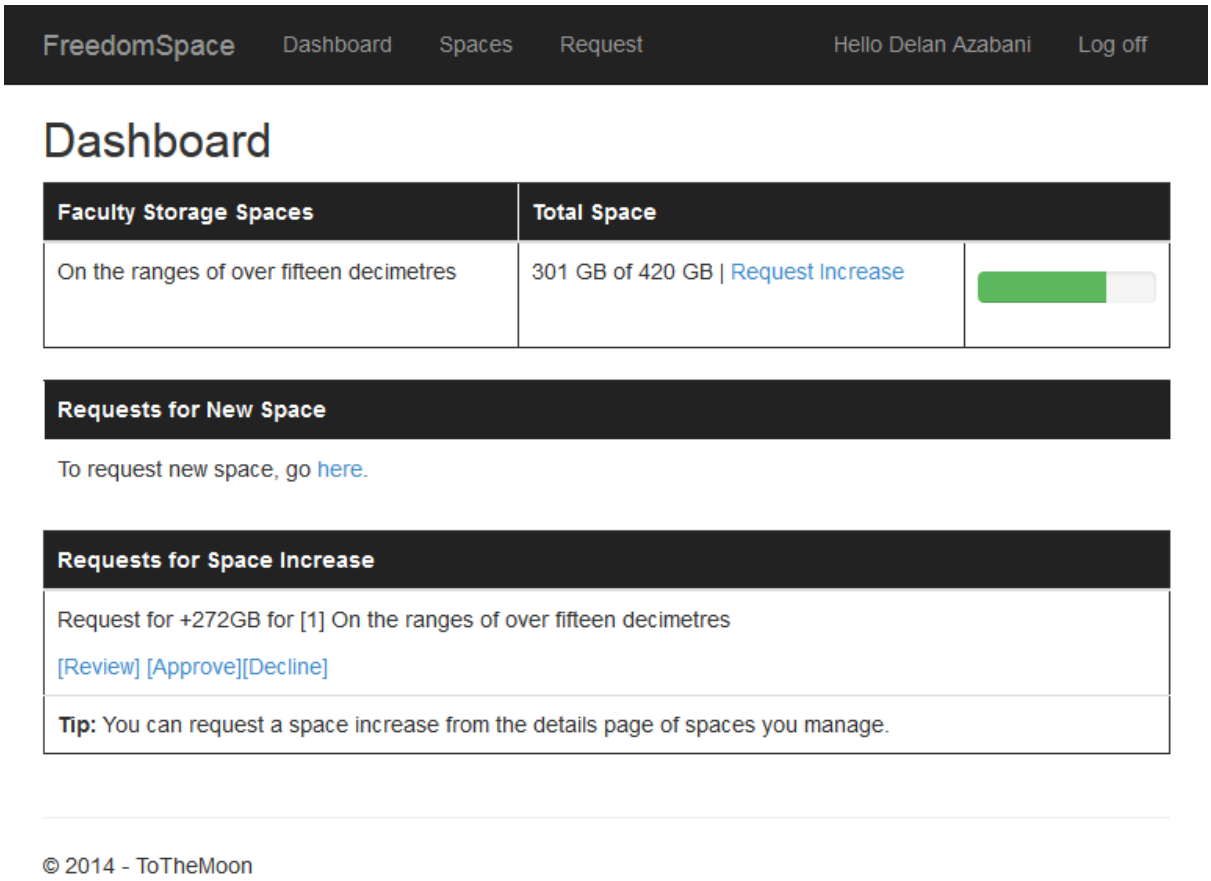


Figure 10: The dashboard as seen by an administrator



- As an approver, I want to review and approve a storage request so that storage can be provisioned.

These four stories are the remainder of the tier one user stories. All four of them relate to storage access; as this system is about managing access to storage for researchers, it is very important that they are completed first.

The first story is about the principal investigator, which is the role of the person who wants to have the storage set for their project. From this first story we can gather that principal investigators want to be able to easily request storage spaces as this is a key part of their role.

The second story can be split into two parts, the first part being that users want to be able to see their storage spaces. This is a critical function of the system because there is no point to having storage spaces if no one can see them. The second part could very easily be forgotten due to the significance of the first part and that is that the users also want to know their access levels for these projects that have space allocated.

The third story gives insight into what is wanted from the system in terms of changing the storage of users. From this we can understand that administrators want to be able to change the permissions, expand and approve storage. This means the system needs to be able to dynamically set storage and user permissions to add in or remove users' access to storage spaces. This was classified as a high priority because it is a core part of the system that access is granted and only to those who are approved.

The fourth story is quite similar to the third in that it is about approving storage requests, which is why it is grouped with it. However it also helps define the roles of administrator and approver, because from this story we can see that only administrators want to be able to change permissions and expand storage, where approvers only want to be able to approve as their name would suggest.

What we learned from this first set stories is that we need to have a well-defined set of rules of user permissions. From the user stories we know that each user has a role within a project and that while some of their roles do overlap or some functions they are all different. These user levels and roles help define what users are allowed to do and access within the system. Due to the key nature of system integrity we decided to come up with a diagram of user permission levels for all the user types. This was used throughout the design phase to create the different user logins in the system.

For the second level of user stories we see many more stories concerning collaboration and changing user access levels for the existing storage spaces.

- As a data manager, I want to grant other researchers access to my storage space so that I can collaborate with others;
- As a data manager, I want to submit additional storage requests so that I can store more research data in my storage space;
- As a principal investigator, I want to grant other researchers access to my storage space so that I can collaborate with others;
- As a principal investigator, I want to grant other researchers data manager level access for my storage space so that they can submit administrative requests; and

- As a principal investigator, I want to submit additional storage requests so that I can store more research data in my storage space.

From the first two stories we see a new user role, the data manager, which is a user who controls the data and access of the projects. From the first of these stories we can see that the data managers need to be able to control who has access to their storage spaces which they want to use for collaboration. We also see that in the third story the principal investigators want to be able to grant other researches access to their storage spaces as well.

This pattern of data manager and principal investigator having very similar stories continues again when we see that they both want to be able to request more storage space for their projects. While most storage requests were in the first priority group of user stories these ones are in the second priority group because they are requesting additional storage which isn't a critical feature.

The fourth story provides a good insight into the power of the principal investigator as they want to be able to create data managers from researchers. This functionality is important because it allows the system to be mostly run by users in their own project rather than having the global administrators set roles for each project which could become a large amount of work if the system is used across a large user base.

There were two more user stories that were set as priority two.

- As an administrator, I want to change the principal investigator for a storage space to replace the existing principal investigator; and
- As an approver, I want to view a list of all storage requests so that I am aware of all requests by principal investigators in my faculty.

From this first story we see that administrators need to have the power to change principal investigators of storage spaces. This is important because if a principal investigator doesn't do their job properly it could be damaging to the project. This also helps us better define the role of the administrators by giving more insight into their needs.

The second story helps us understand the position of approvers within the company because of they want to be able to see faculty based requests we can understand that approvers would be a high position within the faculty and need to see what researchers within the faculty need. This also helps show us that the user classes we create must have a faculty identifier somewhere so that this story can be met.

The third and final group of user stories were the stories we decided were the lowest priority to complete in the system. These were features and functions that did not have much of an impact on the functionality but were more for ease of use.

- As a data manager, I want to revoke an existing researcher's access to my storage space so that they can't access my data;
- As a principal investigator, I want to revoke an existing researcher's manager level access for my storage space so that they cannot submit administrative requests;
- As a principal investigator, I want to revoke existing researchers' access to my storage space so that they can't access my data; and

- As a researcher, I want to receive email notification when my access to a storage space is revoked so that I am aware that I can no longer access it.

There were quite a few user stories relating to revoking access of a researcher either to a space or of their administrative powers of that space. We felt that these would definitely not be needed to demonstrate the product in the few sprints and that there would be no point doing before researchers could be added.

These user stories are much straighter forward than the earlier stories because they are relating to simple functions that help with the maintainability of the software. There is not more to be read out of the first three stories other than the different high level role users want to be able to remove researchers from their projects. It is a simple request and not a hard feature to implement which is why it was pushed back to the lowest priority.

The fourth user story in this set is a bit different though because it is from the researcher who wants to be notified about being removed from the storage space. Email notifications were something that we decided did not need to be down until all of the core functionality is in and working properly. So all email and notification stories were placed in to the lowest priority group.

- As a data manager, I want to receive email notifications so that I am aware that requests have been completed;
- As a principal investigator, I want to receive email notifications so that I am aware that requests have been completed; and
- As a researcher, I want to receive email notifications when I am granted access to a storage spaces so that I can access the data.

All of these stories are about email notifications for features that were requested in higher priority stories. From these though we see that the system does need to be able to sync to a mail server to deliver the email and that all users need to register an email with their account.

One of the notification requests however was different, 'As an administrator, I want to send appropriate notifications so that users can start using their storage spaces.' From this story we need to create a way for administrators to be able to send notifications to users. This story in particular did not however mention what kind of notification should be sent so it was up to us to decide how best to send a notification.

- As an administrator, I want to view a list of requests by submission date so that I can review them chronologically; and
- As an approver, I want to view a list of all storage spaces for my faculty so that I can determine storage space allocated for my faculty.

These two stories show that the high level users want to be able to list submission requests and have the sorted in a particular way. This meant that some sort of algorithm would need to be implemented to sort them; while the first story also refers to submission date which none of the others do, this may have not been stored otherwise. Although similar there are other stories that make a note of faculty so it would be a bit more obvious to store which faculty a project is part of.

The final user stories are about requests for storage.

- As a data manager, I want to view the status of my request so that I know its progress;
- As a principal investigator, I want to view the status of my request so that I know its progress; and
- As an administrator I want to add a comment to a request so that I can highlight any special requirements or additional information related to the request.

These stories show that users want to be able to see the status of their requests so they can then follow up with administrators personally if need be. The third story here does ask for something a bit different though and that is the ability to add comments to requests. This does require adding on some extra functionality, however as a group we decided that it was not core functionality and did not need to be put in until all of the core features and functions were added.

For the product backlog we ended up with seven user stories being classified as highest priority, these were the core functions and features of the system. There were six user stories as medium priority which was mostly usability features such as increased user control. The lowest priority group had thirteen user stories although as explained above many of these were the same story from different users and most of them were increased user control or notification requests.

5 Sprint documentation

5.1 Sprint 1 planning

The user stories that were chosen for the first sprint were the user stories with the greatest importance, serving as the metaphorical foundations of the project. To do this we rearranged and categorised the user stories by their importance, access level requirements, and dependencies to yield the tasks which required the lowest level of requirements to complete first. The first sprint should contain everything that would be mandatory in the second and third sprints, namely:

- As an administrator, I want to log in so that I can review and approve requests;
- As a principal investigator, I want to submit storage requests so that I can have access to a storage space for a research project;
- As a user, I want to log in so that I can perform appropriate functions applicable to my role;
- As a user, I want to view a list of my storage spaces so that I know my access level for my research projects;
- As an administrator, I want to approve a request so that storage can be provisioned, expanded or access permissions changed;
- As an approver, I want to log in so that I can approve storage requests; and
- As an approver, I want to review and approve a storage request so that storage can be provisioned.

These user stories are what we collectively prioritised as the most important tasks to complete before any other parts of the project can be undertaken.

The breakdown that we had planned for the project as a whole was to correlate all user stories to a number depending on what required the least functionality as well as which user stories required others to be completed before they could be implemented. By doing this, we managed to organise all of the user stories into three groups, which ideally would be allocated to each sprint. The first sprint would involve everything on the lowest level, which includes logging into the website, viewing spaces and reviewing requests as the users with the highest levels of access permissions.

With this we estimated that it would take approximately two weeks. This would give us enough time for the group to orientate and establish suitable communications between all members. After that, we would be able to work smoothly throughout the project. Within this fortnight, one week will be set aside to create the user login system and have it operational, then three days will be allocated to implement the viewing of lists and reviewing of storage requests. The remaining four days would go towards fixing defects and polishing what has been done thus far, in addition to working on other projects.

5.2 Sprint 1 implementation

Planned tasks:

- Set up project management tools;
- Authentication system;
- Submit storage and additional storage requests;
- View a list of all users' and faculty storage spaces; and
- Approve and review requests.

Completed tasks:

- Set up project management tools;
- Authentication system;
- Submit storage and additional storage requests;
- Redesign of navigation bar; and
- Frontend addition of interleaved links to relevant pages.

Due to most of the group being new to web development, we focused on achieving the functionality of the user stories, and decided to adjust the frontend to match the initial mockups at a later date.

Multiple accounts with no access restrictions were successfully registered using the website, and all were able to submit storage requests as per the requirements specification. We have been testing the website using the latest released of Google Chrome, Mozilla Firefox and Internet Explorer. At this stage the registration screen accepted a username and password, with no requirement for Curtin ID or email address, therefore checking for a valid email address was not required.

The task effort breakdown for the first sprint is illustrated in Figure 13.

5.3 Sprint 1 review

In the first sprint, the user stories decided upon were fairly ambitious. As a group we completed a number of allocated user stories including:

- As an administrator, I want to log in so that I can review and approve requests;
- As a user, I want to log in so that I can perform appropriate functions applicable to my role; and
- As an approver, I want to log in so that I can approve storage requests.

These user stories overlapped in functionality and so were considered as one single task, which was completed by Delan.

- As a principal investigator, I want to submit storage requests so that I can have access to storage space for research projects;
- As a data manager, I want to submit additional storage requests so that I can store more research data in my storage space; and

- As a principal investigator, I want to submit additional storage requests so that I can store more research data in my storage space.

As with the login user stories, the above stories were considered to be one single task due to the similarity in their functionality, and they were completed by Kye.

- As a user, I want to view a list of my storage spaces so that I know my access level for my research projects; and
- As an approver, I want to view a list of all storage spaces for my faculty so that I can determine storage space allocated to my faculty.

This task was not completed in its entirety at this stage but the majority of it had been completed and was ready for presentation. Completion of this task was a joint effort between Jason and Kye.

There were also planned user stories that at this point were not yet complete, such as:

- As an administrator, I want to approve a request so that storage can be provisioned, expanded or access permissions changed; and
- As an approver, I want to review and approve a storage request so that storage can be provisioned.

Unfortunately, due to a combination of insufficient time management, informal role allocation and a general team-wide unfamiliarity with the chosen development environment, these user stories were not addressed.

Additional items completed concerning improvements to user experience include:

- Redesign of navigation bar; and
- Front end addition of interleaved links to relevant pages.

These changes were completed by Luke.

5.4 Sprint 1 retrospective

This sprint was objectively rather productive. Of ten assigned tasks, the group worked together to complete eight requirements that were initially planned, as well as the addition of two unplanned tasks. These additions were vital in order to ensure easy navigation of the project by the end user.

Issues encountered mainly concerned:

- The lack of a clear definition of group member task allocations. This was the most fundamental flaw with our approach to this sprint. Due to a combination of poor communication and job allocation, we had a situation where many group members had accidentally begun work on the same tasks and for this reason, tasks were left untouched while a lot of effort was duplicated and ultimately discarded.
- Many team members at this stage were completely unfamiliar with the chosen working environment of ASP.NET MVC 5. While not quite as important as the previous issue, at this stage, it was definitely a hindrance to be trying to write code in an environment that none of the group members were familiar with.

For these reasons, the group as a whole resolved to be more explicit in task allocations and ownership. Expected task time frames were decided on moving forward, as a preventative measure for tasks being left incomplete.

5.5 Sprint 2 planning

What was designated for the second sprint were the second set of user stories, which involved building up from what should have been accomplished in the first sprint. Ideally, this would mean that authentication and request approvals would have been completed.

The second set of user stories would involve creating a access permissions system whereby — depending on a particular user's access level — the system would provide and/or restrict functionality. The user stories that were chosen for the second sprint include:

- As a data manager, I want to grant other researchers access to my storage space so that I can collaborate with others;
- As a data manager, I want to submit additional storage requests so that I can store more research data in my storage space;
- As a principal investigator, I want to grant other researchers access to my storage space so that I can collaborate with others;
- As a principal investigator, I want to grant other researchers manager level access for my storage space, so that they can submit administrative requests;
- As a principal investigator, I want to submit additional storage requests so that I can store more research data in my storage space;
- As an administrator, I want to change the principal investigator for a storage space to replace the existing principal investigator; and
- As an approver, I want to view a list of all storage requests so that I am aware of all requests by principal investigators in my faculty.

Assuming the aforementioned user stories are completed by the end of the sprint, the project should also be able to handle the next set of user stories to be implemented. This is because the second sprint is purely intended to ensure that access levels and data storage function correctly.

From the user stories outlined above, the second level of the breakdown is found, including the tasks that require logging in as a user with a specific user access level. This involves submitting and granting storage space requests for 'middle level' users.

The time estimated for this sprint would again be two weeks, with the granting of storage requests taking one week. The additions to the prototype that handle request submission should only take three days at most to implement, and the remaining days would be allocated to the ability to view and change the principal investigator of storage spaces.

5.6 Sprint 2 implementation

Planned tasks:

- View a list of all storage requests;
- Grant other researchers access to my storage spaces;
- Approve and review storage requests;
- User access levels and privileges; and
- UML diagram.

Completed tasks:

- Dashboard frontend with static HTML;
- Cancellation and deletion of space requests;
- Recorded timestamps for each request; and
- UML diagram.

In an earlier meeting Callum suggested using the Decorator software pattern to implement the different user roles as they have subset permissions. The group agreed to research the Decorator pattern in order to use it. However, due to the complexity of the MVC5 framework and our lack of experience using it, it was decided that the pattern was too complex to implement using a foreign framework.

We then decided our account level types will be implemented as different viewing permissions. This allows us to base what the user is allowed to see on their role as per the requirement specification. Although, the tradeoff is that the website is not as secure as would be preferred. Due to the time constraints the group decided to pursue the safer option of fulfilling the required functionality with lesser security measures.

The task effort breakdown for the second sprint is illustrated in Figure 14.

5.7 Sprint 2 review

At the epoch of the second sprint, user stories decided on were scaled back quite drastically to accommodate incomplete items in the sprint backlog. Of the chosen user stories, the following were completed:

At this stage in the project, there were a worrying number of incomplete user stories now moved into our unfinished item backlog, such as:

- As an approver, I want to review and approve a storage request so that storage can be provisioned;
- As an approver, I want to view a list of all storage requests so that I am aware of all requests by principal investigators in my faculty;
- As a data manager, I want to grant other researchers access to my storage space so that I can collaborate with others;
- As an administrator, I want to approve a request so that storage can be provisioned, expanded or access permissions changed; and

- As an approver, I want to review and approve a storage request so that storage can be provisioned.

Of these, the most important functionality by far were the user access levels, and to not yet have this complete took a very noticeable cleave out of the team's morale.

In addition to the planned functionality to be worked on in this sprint, there were also a number of features added supplementary to what was planned:

- Dashboard front end using static HTML.

As completed by Jasmine, this was an invaluable feature to have going into the presentation for the second sprint as it was the first major step towards transposing our initial mockups and planning into an actual product.

- Users being able to cancel/delete their own space requests; and
- Requests now storing a timestamp to reflect when they were made.

These functionalities were implemented by Kye.

5.8 Sprint 2 retrospective

Aspects that worked well at this point were unfortunately rather scarce.

Over the course of the sprint, many obstacles became apparent:

- Members were sick due to the time of the year. While it is unavoidable that this can happen, the team unfortunately had made no provisions to handle the situation. When people were unable to complete their tasks, there was little and rather poor communication within the group to attempt to overcome the issue.
- Sprint backlog items were not adequately addressed. Over the course of this sprint, the team morale reached a global minimum. Consequently team members felt uncompelled to address any tasks which resulted in a less than desirable yield.

After what was undeniably the worst sprint so far, the team as a whole devised stricter guidelines for the next sprint in order to produce more consistent results going forward. Guidelines considered of utmost importance were:

- Motivate team members more effectively to stick to designated tasks and time schedules; and
- Issues that arise will be brought up both in person as well as the Facebook group in order to alert all group members of potential problems as soon as they develop.

5.9 Sprint 3 planning

By the third sprint hopefully, the previous user stories would have been completed and implemented correctly. The rest of the user stories are small additions onto the already existing system, so even though there are a vast quantity more user stories in the sprint than others, these user stories all function in a similar fashion to one another.

The user stories selected to be implemented in the third sprint were:

- As a data manager, I want to receive email notifications so that I am aware that requests have been completed;
- As a data manager, I want to revoke existing researchers' access to my storage space so that they can't access my data;
- As a data manager, I want to view the status of my request so that I know its progress;
- As a principal investigator, I want to receive email notifications so that I am aware that requests have been completed;
- As a principal investigator, I want to revoke an existing researcher's manager level access for my storage space so that they cannot submit administrative requests;
- As a principal investigator, I want to revoke existing researchers' access to my storage space so that they can't access my data;
- As a principal investigator, I want to view the status of my request so that I know its progress;
- As a researcher, I want to receive email notification when I am granted access to a storage space so that I can access the data;
- As a researcher, I want to receive email notification when my access to a storage space is revoked so that I am aware I can no longer access it;
- As an administrator, I want to add a comment to a requests so that I can highlight any special requirements or additional information related to the request;
- As an administrator, I want to send appropriate notifications so that users can start using their storage spaces;
- As an administrator, I want to view a list of requests by submission date so that I can review them chronologically; and
- As an approver, I want to view a list of all storage spaces for my faculty so that I can determine storage space allocated for my faculty.

The time available would be three weeks, and this would include finishing the formatting and patching all remaining defects, as well as ensuring that the system runs in accordance with the client's requirements.

The email notification module would require three days, and viewing the status of requests would require a further four days to implement. The second week would be dedicated to revocation of storage space access, as this involves changing the access control data between a user and a space, rather than the access role of a user.

5.10 Sprint 3 implementation

Planned tasks:

- Viewing a list of all storage requests;

- Approve and review storage requests;
- User access levels and privileges;
- Granting access to storage spaces;
- Granting manager level access to storage spaces;
- Sending appropriate notifications;
- Receiving email notifications;
- Viewing status of requests;
- Revoking access to storage spaces; and
- Adding comments to requests.

Completed tasks:

- Viewing a list of all storage requests;
- Approve and review storage requests;
- User access levels and privileges;
- Granting access to storage spaces;
- Granting manager level access to storage spaces;
- Sending appropriate notifications;
- Receiving email notifications;
- Viewing status of requests;
- Revoking access to storage spaces; and
- Adding comments to requests.

As the project is not being hosted online, we are unable to actually send email notifications. Instead, we are working around this limitation with a page that appears when an email notification would have been sent that will still have the same content and data.

The task effort breakdown for the third sprint is illustrated in Figure 15.

5.11 Sprint 3 review

It was near the end of the second sprint where the team didn't realise how close the deadline was to the final presentation of the project, and it was only then did the group finally shift gears and come together to start finishing off the user stories rather quickly. With efforts from Kye and Luke, the user access levels were quickly finished and with that, the following list of user stories were also completed with it, from the backlog of sprint two:

- As an approver, I want to review and approve a storage request so that storage can be provisioned;

- As an approver, I want to view a list of all storage requests so that I am aware of all requests by principal investigators in my faculty;
- As a data manager, I want to grant other researchers access to my storage spaces so that I can collaborate with others;
- As an administrator, I want to approve a request so that storage can be provisioned, expanded or access permissions changed; and
- As an approver, I want to review and approve a storage request so that storage can be provisioned.

When this was completed, Jason in addition to Luke and Kye worked tirelessly to get the project back onto a reasonable pace, and could possibly complete it on time. These were the remaining user stories that required work:

- As a data manager, I want to revoke existing researchers' access to my storage spaces so that they can't access my data;
- As a data manager, I want to view the status of my request so that I know its progress;
- As a principal investigator, I want to receive email notifications so that I am aware that requests have been completed;
- As a principal investigator, I want to revoke an existing researcher's manager level access for my storage spaces so that they cannot submit administrative requests;
- As a principal investigator, I want to revoke existing researchers' access to my storage spaces so that they can't access my data;
- As a principal investigator, I want to view the status of my requests so that I know its progress;
- As an administrator, I want to send appropriate notifications so that users can start using their storage spaces;
- As an administrator, I want to view a list of requests by submission date so that I can review them chronologically; and
- As an approver, I want to view a list of all storage spaces for my faculty so that I can determine storage space allocated for my faculty.

5.12 Sprint 3 retrospective

Our team eventually managed to coordinate efforts relatively smoothly, finishing the prototype — and more broadly, the project — on time, although the stage at which this occurred was significantly later than what would be optimal.

The vast majority of the user stories provided by the client were implemented in this final sprint, in an effort to count the subpar productivity yielded by the former two sprints. By this time in the semester however, the ability of group members to find enough time to complete the shared workload became significantly more difficult, which bolstered the fragility of our commitment to the project deadline.

5.13 Details of the design

After much deliberation over the user stories and the permissions model, our team devised a set of models and their relationships as defined by the UML and entity-relationship diagrams in Figure 11 and Figure 12 respectively.

It is apparent that there is much in common between the model classes `NewSpaceRequest` and `IncreaseSpaceRequest`. While we initially tried to refactor this by making them subclasses of an abstract parent class `Request`, this conflicted with the `EntityFramework` object-relational mapper that is an integral part of ASP.NET MVC 5. The ORM would return instances of the parent class, which would require casting, then fail to store in the database when changed. This impasse forced us to allow the small amount of code duplication required to eliminate the inheritance relationship in our mapped classes.

Figure 11: The Unified Modeling Language diagram for this prototype

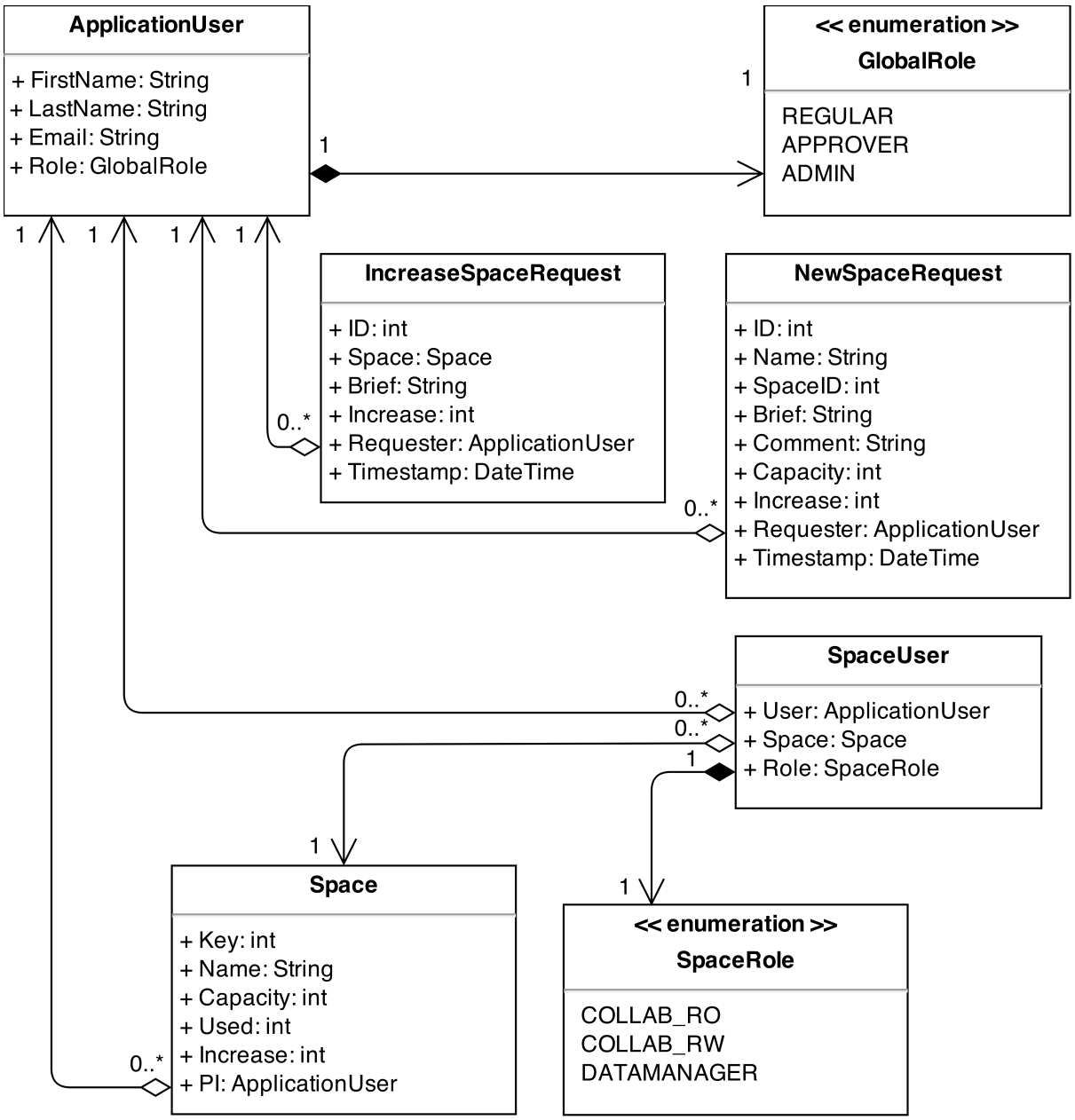


Figure 12: The entity-relationship diagram for this prototype

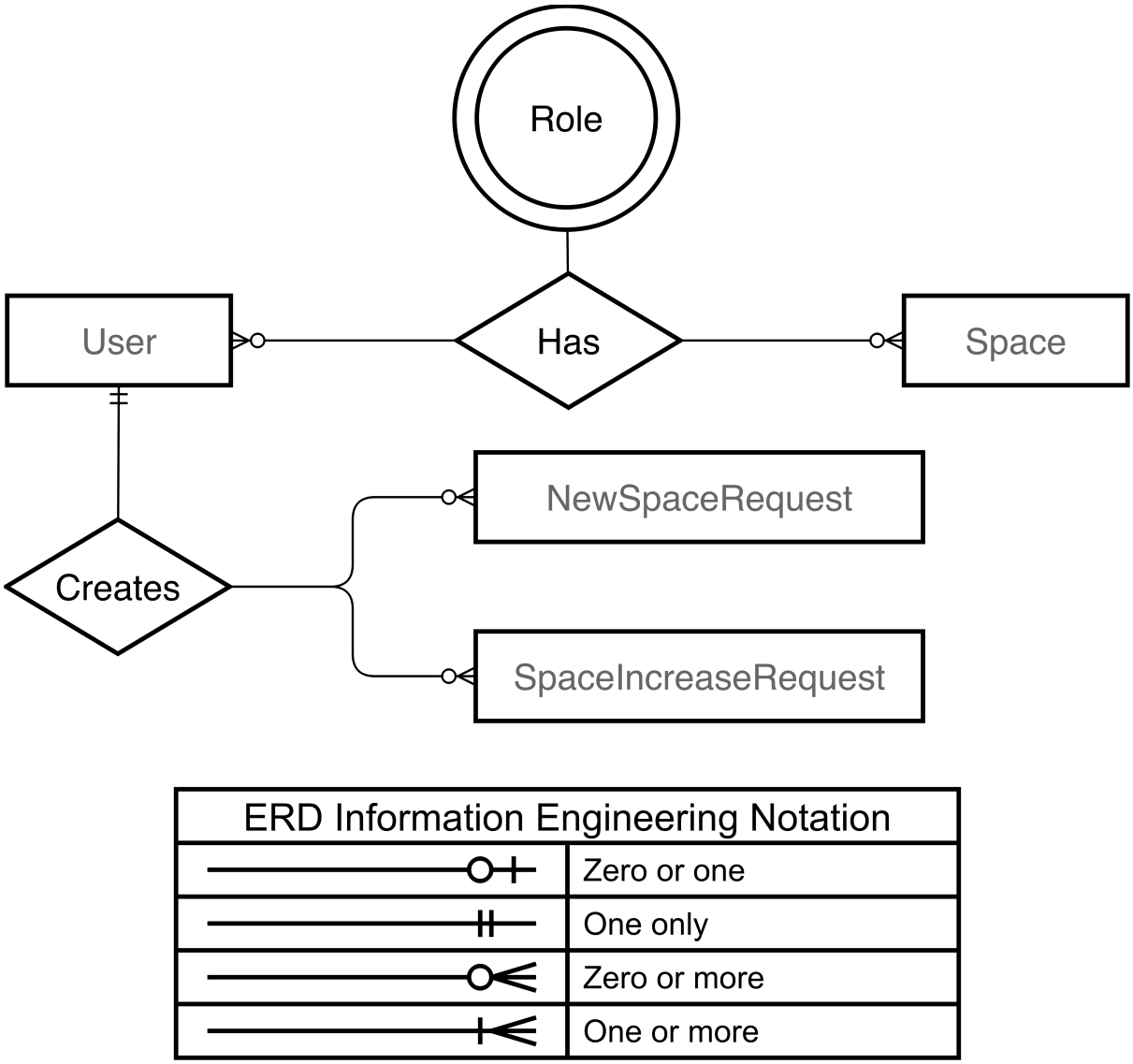


Figure 13: The task effort chart for the first sprint

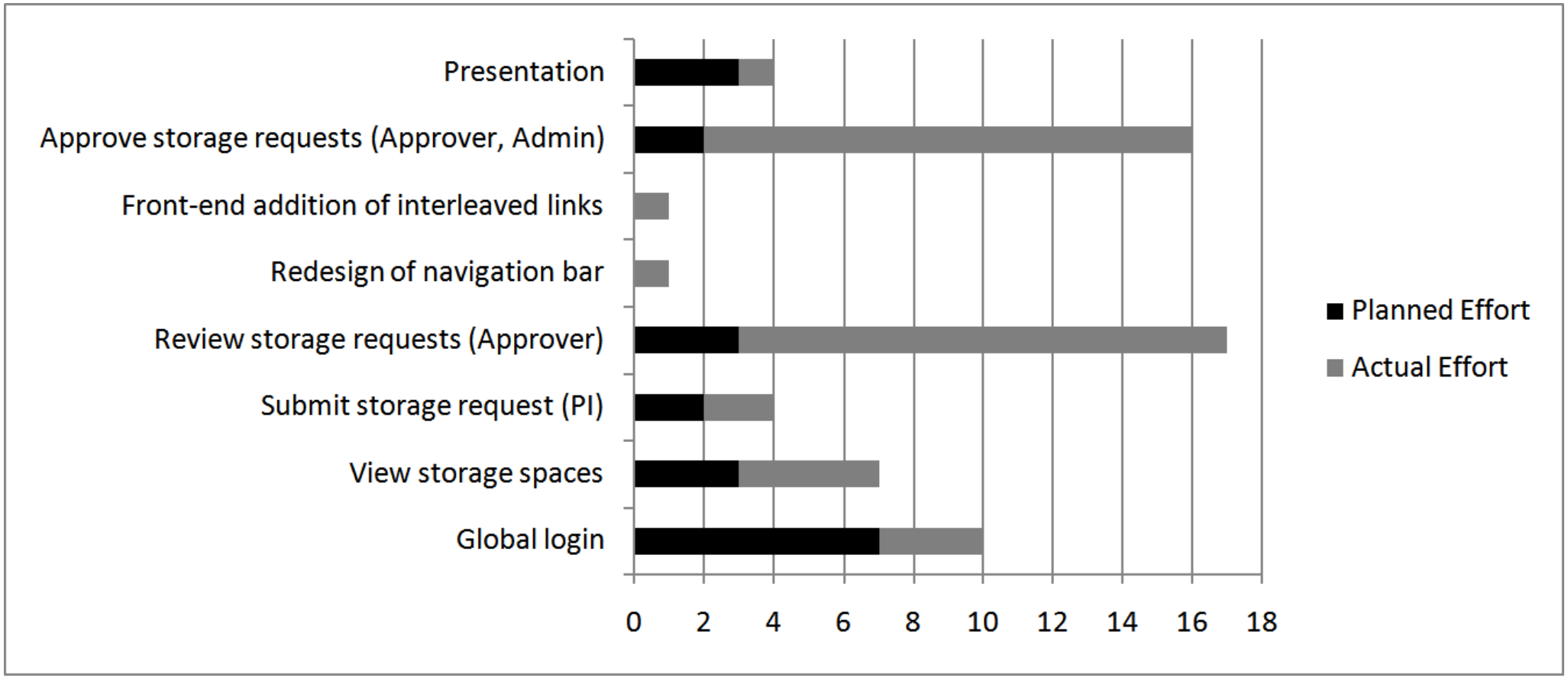


Figure 14: The task effort chart for the second sprint

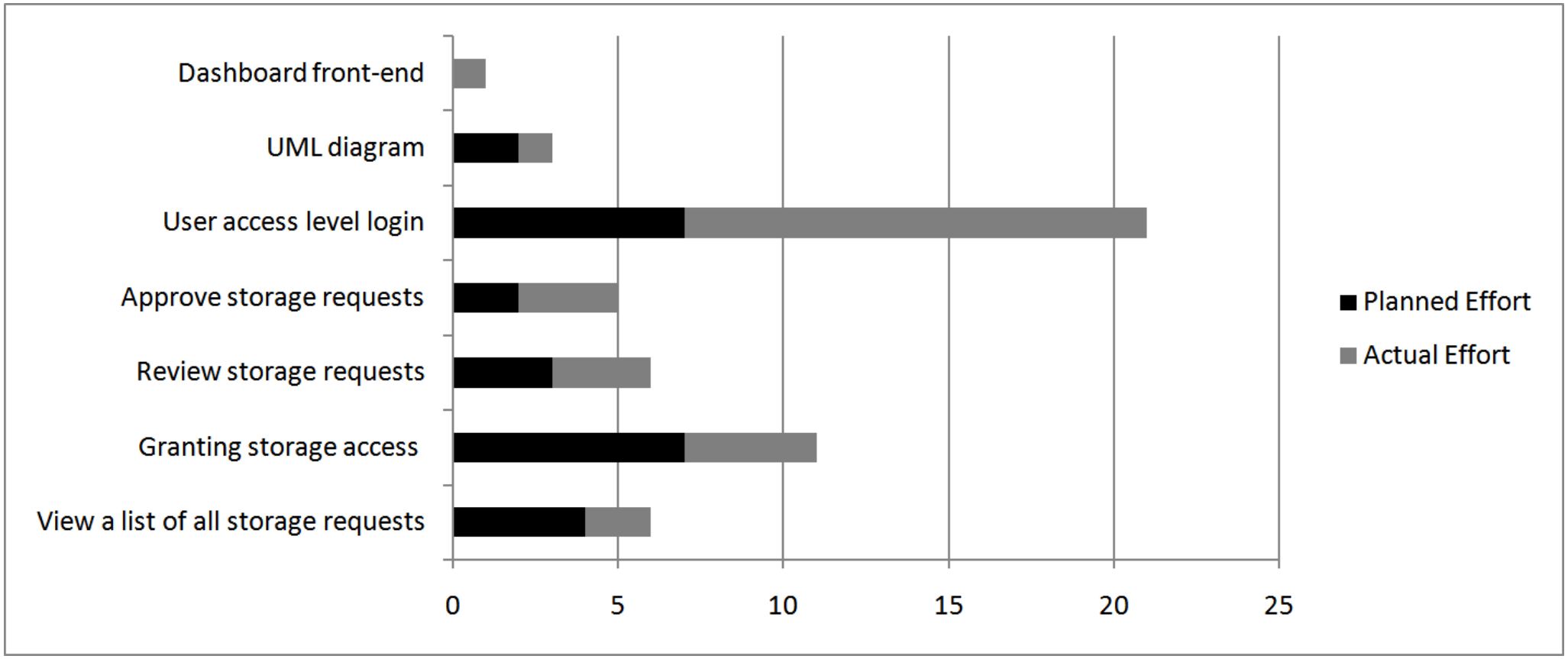
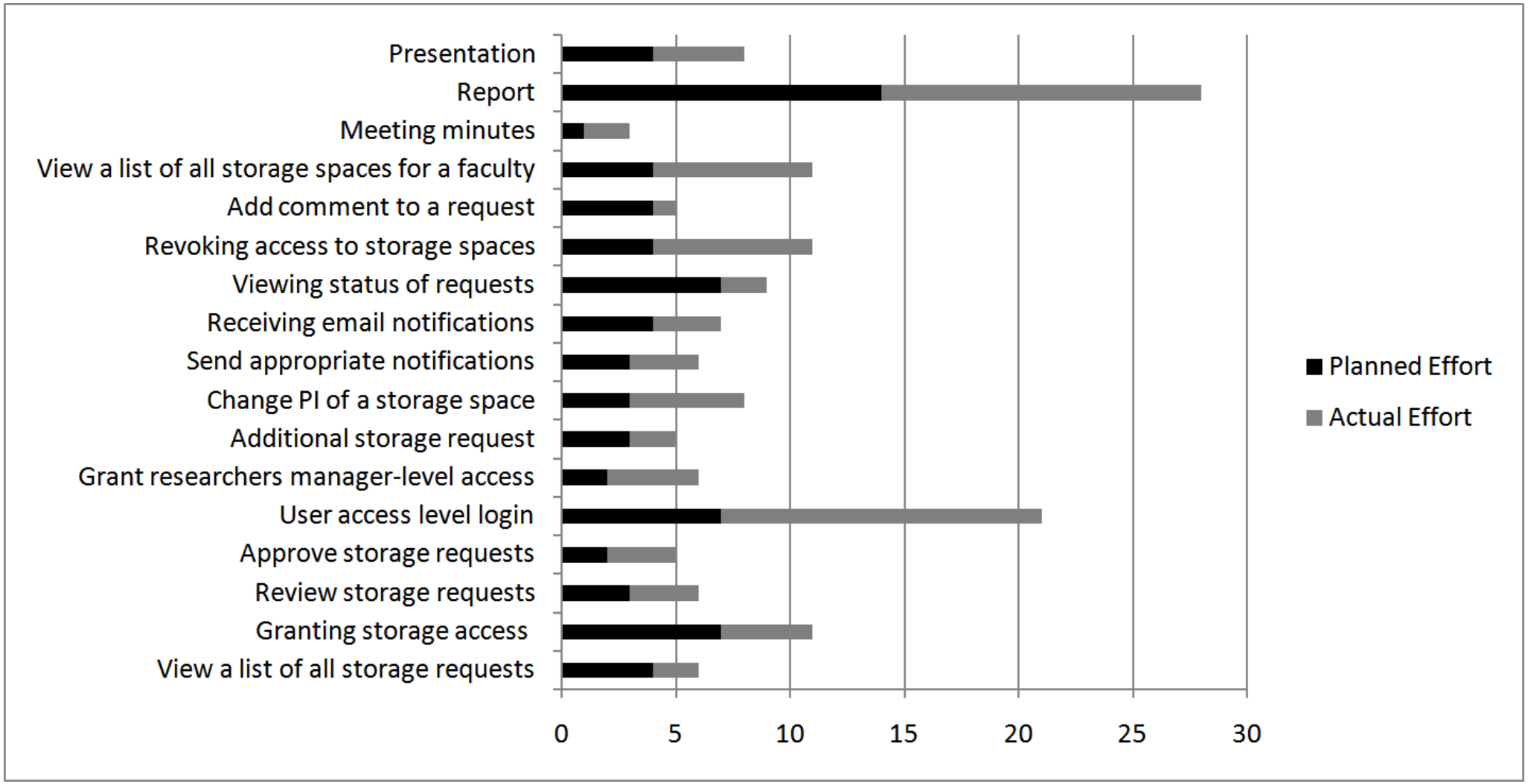


Figure 15: The task effort chart for the third sprint



5.14 Minutes for Scrum meeting 1

Date: 2014-04-03

Start time: 4:00 p.m.

Discussing how the project should be completed and the technologies we are using to implement the prototype.

- Front end: Bootstrap;
- Back end: ASP.NET MVC 5;
- Database: Microsoft SQL Server;
- Time logging: Google Docs form;
- Number of sprints: three.

Front end tasks to complete during the first sprint:

- Requests pages and forms;
- Login screen; and
- Notification space.

End time: 5:00 p.m.

5.15 Minutes for Scrum meeting 2

Date: 2014-04-15

Start time: 12:10 p.m.

Meeting for discussing what is functioning in the first sprint, and what needs to be added during sprint two.

The list of user stories that were completed were:

- Submission of storage requests; and
- Users being able to log in.

Tasks that were partially completed were:

- Administrators being able to approve and change permissions; and
- Principal investigators submitting storage requests for research.

All of the user stories were completed on the front end load where users could see what it looked like, but all of the functionality was yet to be finished. These will need to be completed within the next sprint because there were issues with access levels of the different types of users, and we were unsure how to handle them until we completely designed the access levels for all the users, which would be done in the second sprint.

End time: 1:00 p.m.

5.16 Minutes for Scrum meeting 3

Date: 2014-04-25

Start time: 1:03 p.m.

Task allocation:

- Robert: UML diagram, user stories, list of back end and front end design decisions, implementation and sprint review;
- Jasmine: high level task breakdown, effort estimation;
- Callum: implementation of approvers' user stories;
- Delan and Luke: implementation of granting and revocation of permissions for data managers;
- Kye: implementation of user authentication and access levels; and
- Jason: implementation of the permissions list view.

All further issues will be raised on the Facebook group, so that all members can have the opportunity to assist. The original plan from earlier sprints was too complicated, and it will be modified to make it simpler. Approvers will be managed with a flag at the user level, indicating whether they are allowed to approve requests or not.

The UML diagram was designed for the implementation.

End time: 2:34 p.m.

5.17 Minutes for Scrum meeting 4

Date: 2014-05-01

Start time: 4:00 p.m.

Meeting for discussing what is functioning in sprint two and what needs to be added to sprint three, as well as allocation of report writing tasks to group members. The user stories that were planned for full implementation were for data managers and principal investigators to be able to submit requests, and to allow privileged users to grant and revoke access to storage spaces.

The list of user stories that were completed were:

- Submission of storage requests for data managers and principal investigators;
- Viewing a list of storage spaces for approvers;
- Administrators being able to approve and change permissions; and
- Principal investigators submitting storage requests for research.

User stories that were partially completed were:

- Administrators being able to change the principal investigator on a storage space; and

- Principal investigators granting researchers read, write and data manager level access to storage spaces.

As a team, the decision was made to shift a majority of the focus towards the documentation of the sprints and the report. The report will be created for global editing within the members to update and edit where possible. We will finish the demonstration for displaying research by the next week. Jasmine has been assigned to conduct the final presentation. Callum and Robert have been assigned to write the 'details of design' and 'Scrum meeting' sections of the final report. Luke and Delan were assigned to completing the backend models and controllers, while Jason and Kye are to finish developing the views and the user interface. Callum shall polish the finished project for presentation and any additional documentation within the report.

End time: 4:37 p.m.

5.18 Minutes for Scrum meeting 5

Date: 2014-05-15

Start time: 3:59 p.m.

By the end of sprint three, we had assumed that everything for the project would be functional, except for some bugs which are to be expected.

The list of user stories that were completed were:

- Administrators being able to change the principal investigator for a storage space;
- Principal investigators granting researchers read, write and data manager level access to storage spaces;
- Reception of email notifications as any privileged user;
- Revocation of access as any user;
- Addition of comments to requests;
- Viewing a list of request sorted by submission date; and
- Viewing a list of storage spaces for a specific faculty.

Tasks that were partially completed were:

- Formatting for the user interface.

Report task reallocation:

- Callum: product backlog;
- Delan: details of design, conclusion and summary;
- Jasmine: background, implementation details;
- Jason: abstract, introduction and objective;
- Kye: project review;
- Luke: sprint review and retrospective; and

- Robert: sprint documentation and planning.

Callum and Jasmine will edit and correct the final report copy, while Delan will complete the formatting and typesetting of the report.

End time: 4:24 p.m.

5.19 Minutes for Scrum meeting 6

Date: 2014-05-26

Start time: 9:11 a.m.

Finalisation of report, documentation and presentation.

The group shall have the report finished by Friday for editing and formatting. Delan will create the PowerPoint file for the final presentation. Depending on Aneesh's answer, everyone could be presenting, or only some group members.

End time: 9:21 a.m.

5.20 Minutes for Scrum meeting 7

Date: 2014-05-29

Start time: 4:06 p.m.

Last meeting before the final presentation.

Contact Delan when finishing report segments so formatting can be completed. Jasmine is unwell; if she is unable to present, duties will be passed onto Callum, Robert or Luke depending on the day of the presentation. Small changes will be made to the prototype.

End time: 4:27 p.m.

6 Project review

There were many lessons learned from working on this project using the Scrum process. We did quite a few things right with our process. In our initial group meetings we sorted out our product backlog right away which really gave us a good idea of what we needed to do. We also split the work load up into the three sprints and set our sprint goals. Our meeting organisation was quite well done given the size of the group and the available time everyone had given their respective units, we were always able to have weekly meetings even over the break.

Initially the group was quite motivated and quite eager to get started however the exact work of each group member was not clearly defined and the team was not communicating as much as they need to be for using a process like Scrum.

While the group was meeting weekly, these meetings were often missing a couple of members, it was not having daily Scrum meetings and the group was not communicating about the project to each other outside of the weekly meetings. This led to a lot of work being done unnecessarily. Another major problem facing the group was the unfamiliarity with ASP.NET MVC, only one member had previously used it and the rest were learning as they were going which led to some members work being of an unacceptable low quality.

All of these issues led to the group becoming very unmotivated for the time of the second sprint. This was also the same time other units had begun to really require time for their assignments. During this time communication was at an all-time low and the members did not know what members were doing. During this time very little work was done.

Another major problem was the lack of a Scrum master for the group. Without a Scrum master the group did not really know how to follow the process and it resulted in work not being done and time for work not being logged. If we had closely followed Scrum like we were supposed to many of our problems would not have occurred however no-one in our group had a very good understanding of Scrum.

Throughout this project we learned quite a few lessons. The most important is having a smaller group, with a process like Scrum more people only makes it harder and in a university setting trying to have a meeting with 7 people is quite hard. We learned that we need to not get unmotivated as easily otherwise it results in a lot more stress and rushed work down the line.

For future students we would definitely recommend maintaining a small group rather than thinking a large group would mean less work overall. The amount of work in taking to work well together as a team increases a lot for each new person added in to the project.

7 Conclusions and summary

In the last three months, our team *To the Moon* has conceived, designed, developed, tested and delivered a functional prototype for a storage management system that could be used in a university setting. We strived to adopt as many industry standard methodologies, paradigms and technologies as we could, including the Scrum method, the model-view-controller pattern and distributed version control.

Many factors introduced tangible challenges against the efficacy of our teamwork, such as the coordination of communication amongst a diverse team of seven members, a hostile time environment where this project must be balanced with other unrelated assessements and an unavoidable unfamiliarity with some of the technologies involved.

The overall problem that needed to be solved was the scalability of a software development team. Of course, no team in existence scales in a purely linear manner against the number of members, but it is an ideal that we must all strive for, as non-trivial software projects that can be developed completely by an individual are scarce.

For a significant portion of this group's members, *Project Design and Management 300* served as our first group software project, serving as a marked change from histories of individual hobby projects and educational assessments. As such it is understandable that the project did not flourish as smoothly as we would have hoped.

To round this report off, the following are a summary of the lessons learned:

- Strive to organise *regular* meetings that include *all* group members, whether or not they are conducted in person or online;
- Allocate roles formally and strictly to minimise duplication of effort;
- Place a greater emphasis on working with technologies that the most group members are familiar with, when opposed to ideological purity;
- Make clear and realistic provisions to handle the illness of team members;
- Ensure that developers remain constantly motivated throughout the project; and
- Clarify any ambiguous components of user stories and other client documentation as soon as possible.

References

37signals, LLC. 2004. *Basecamp Launches (Signal vs. Noise).* http://signalvnoise.com/archives/000542.php.

Apache Friends. 2014. *XAMPP Installers and Downloads for Apache Friends.* https://www.apachefriends.org/.

Django Software Foundation. 2014. *The Web framework for perfectionists with deadlines.* https://www.djangoproject.com/.

Facebook, Inc. 2014. *Facebook groups.* https://www.facebook.com/about/groups.

Fog Creek Software, Inc. 2014. *Trello.* https://trello.com/.

GitHub, Inc. 2014. *GitHub — Build software better, together.* https://github.com/.

Google Inc. 2014. *Using Google Drive — Learn about new features.* https://www.google.com/drive/using-drive/.

iVEC. 2013. *Data Storage/Accessing the Data Store - iVEC User Documenation [sic].* https://portal.ivec.org/docs/Data_Storage/Accessing_the_Data_Store.

JGraph Limited. 2014. *draw.io.* https://www.draw.io/.

Microsoft Corporation. 2014. *ASP.NET MVC 5 : The Official Microsoft ASP.NET Site.* http://www.asp.net/mvc/mvc5.

Ronacher, Armin. 2014. *Welcome — Flask (A Python Microframework).* http://flask.pocoo.org/.

Twitter, Inc. 2014. *Bootstrap.* http://getbootstrap.com/.