BCIS Project Proposal

# UpStage

# Alyssa Byun, Joshua Kartono, Jing Han

Version 2.0

Contents

[UpStage 0](#_Toc427699315)

[Alyssa Byun, Joshua Kartono, Jing Han 0](#_Toc427699316)

[1. Contact information 3](#_Toc427699317)

[1.1Students 3](#_Toc427699318)

[1.2 Supervisor 3](#_Toc427699319)

[1.3 Client 3](#_Toc427699320)

[2. Term of Reference 4](#_Toc427699321)

[Introduction 4](#_Toc427699322)

[3. Rationale for the project 4](#_Toc427699323)

[3.1 The existing system 4](#_Toc427699324)

[3.2 Why this project is needed? 5](#_Toc427699325)

[4. Skills and knowledge involved 5](#_Toc427699326)

[4.1 Personal Capabilities 5](#_Toc427699327)

[4.2 Professional Capabilities 6](#_Toc427699328)

[4.3 Technical Capabilities 7](#_Toc427699329)

[5. Scope and Objectives 7](#_Toc427699330)

[6. Project Approach 8](#_Toc427699331)

[6.1 Project Methodology 8](#_Toc427699332)

[6.1.1 Agile 8](#_Toc427699333)

[6.1.2 Methodology 9](#_Toc427699334)

[6.1.3 Rationale 9](#_Toc427699335)

[6.1.4 Project Tasks 10](#_Toc427699336)

[6.1.5 Project Deliverables 11](#_Toc427699337)

[6.2 Alternative Methodologies 11](#_Toc427699338)

[6.2.1 Scrum 11](#_Toc427699339)

[6.2.2 Feature Driven Development 12](#_Toc427699340)

[6.2.3 Test Driven Development 12](#_Toc427699341)

[6.3 Project Development Tools 12](#_Toc427699342)

[6.3.1 Front-End Options 12](#_Toc427699343)

[6.3.2 Back-End Options 13](#_Toc427699344)

[6.3.3 Architecture Pattern Options 13](#_Toc427699345)

[7. Project Plan 14](#_Toc427699346)

[7.1 Initial Project Schedule 14](#_Toc427699347)

[7.2 Quality Assurance 15](#_Toc427699348)

[7.2.1 Quality Assurance Plan 15](#_Toc427699349)

[7.2.2 Quality Assurance Tools 16](#_Toc427699350)

[7.3 Review Plan 16](#_Toc427699351)

[7.4 Estimate all costs incurred 16](#_Toc427699352)

[7.4.1Student Costs 16](#_Toc427699353)

[7.4.2 University Costs 17](#_Toc427699354)

[7.4.3 Client Costs 17](#_Toc427699355)

[7.5 Communication Plan 17](#_Toc427699356)

[7.5.1 Method for updating the communications management plan 19](#_Toc427699357)

[7.5.2 Escalation procedures 19](#_Toc427699358)

[7.6 Risk Management Plan 20](#_Toc427699359)

[7.7 Definition of Done 22](#_Toc427699360)

[8. Disclaimer 23](#_Toc427699361)

[9. Glossary 24](#_Toc427699362)

[10. Appendix 25](#_Toc427699363)

[10.1 Product Backlog 25](#_Toc427699364)

[10.2 Team Contract 29](#_Toc427699365)

[11. References 30](#_Toc427699366)

# Contact information

## 1.1Students

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Student ID | Major | Phone | Email |
| William Stokes | 1390780 | Software development | 0223919890 | crusantriaa@gmail.com |
| Siatua Uili | 1135311 | Software development | 0211569048 | siatua.uili@gmail.com |
| Alyssa Byun | 0950756 | Software development | 021790017 | missbyun@hotmail.com |
| Joshua Kartono | 1380483 | Software development | 0223980260 | joshua.kartono77@gmail.com |
| Jing Han | 1380079 | Software development | 0212644996 | hjwwj1987@gmail.com |

## 

## 1.2 Supervisor

|  |  |  |
| --- | --- | --- |
| Name | Email | |
| Anne Philpott | anne.philpott@aut.ac.nz |

## 

## 1.3 Client

|  |  |
| --- | --- |
| Name | Email |
| Vicki Smith | digitalsmith@clear.net.nz |
| Helen Varley Jamieson | helen@creative-catalyst.com |

# ****Term of Reference****

## ****Introduction****

**UpStage is an open source web-based platform. It allows registered users to perform through** images, animations, audio, web cams, text and drawing, as well as interact in real time with their audience.(UpStage, n.d.)

In 2004, when UpStage was created, the major requirement was accessibility and ease of access (that is no downloads required and that it worked via limited bandwidth). UpStage is no longer compatible with current technologies specifically working on mobile devices and tablets because it uses Flash which renders it inaccessible to an increasing number of potential audience members.

This project will consist of developing the new version of UpStage based on how the clients currently use it and implementing some additional functions while maintaining the existing UpStage in the interim. The new version of UpStage should fulfill the objective.

# Rationale for the project

## 3.1 The existing system

Since UpStage has been developed by approximately 40 AUT students over many years, the functionalities for this application have been improved. While it provides users more functionality, the internal structure of this application became untidy and broken. Therefore, it is difficult for UpStage developers to read and understand the code written by the past developers, and also it is difficult to extend the code further as additional code will make it unclean and cause bugs and instability in the software.

Another issue that the existing application has is that it is not operational on mobile devices. Since the smartphones and tablets were introduced and steadily growing, the software that is easy to be accessed by mobile devices is expected and preferred by users. However, the UpStage uses Flash and Flash does not work on mobile devices.

## 3.2 Why this project is needed?

The development team believes that there needs a development of a new UpStage as we will not be able to extend the existing application. Continuous developing on the existing UpStage will increase the possibility for more malfunctions. The new UpStage should be defect free. If the known defects of the existing application were the errors that could be easily fixed to enable it to perform as a new system, we would choose to just maintain and repair the existing structures. However, the development of a new UpStage should resolve all the biggest issues and problems of existing UpStage. During the development of the new UpStage any aspects which benefit maintaining the existing version might be applied, but the main focus will be on developing a new software that reflects the needs of the uses and builds on the possibilities of current coding languages.

The new UpStage must be still web-based and provides the same functionality but it should be a mobile-friendly application. The spaghetti code of the existing system must not be reused. The internal structure should be tidily redesigned.

# Skills and knowledge involved

## 4.1 Personal Capabilities

|  |  |  |  |
| --- | --- | --- | --- |
| Skill | Description | Presence in Team | Time for Upskilling |
| **Research** | The ability to find the correct methods to approach the issues or problems in the project. | Yes | Documentations |
| **Problem Solving** | The ability to deal with problems or challenges. | Yes | Browsing Internet |
| **Communication** | The ability to deliver the knowledge and information to other people. | Yes |  |
| **Quick Learning** | The ability to understand the existing code that is written by other developers. | Yes |  |
| **Debugging** | The ability to identify and remove errors or defects from the software. | Yes | While working on the maintenance of existing UpStage |
| **Software Architecture and Design** | The ability to construct a good internal structure | No | Software design phase |
| **Time Management** | The ability to manage time to ensure that the work is done on time following the schedule and not cause any delays due to overloading. | No |  |

## 4.2 Professional Capabilities

|  |  |  |  |
| --- | --- | --- | --- |
| Skill | Description | Presence in Team | Plan for Upskilling |
| **Teamwork** | Teamwork is every team members contributing to make a valuable product. Understanding of other members’ current progress is required. | Yes | Through the whole life cycle of this team project |
| **Knowledge of ‘Agile’** | As the ongoing project team has already been using Agile methodology for developing this software, the team should have enough understanding of this methodology. | Yes | Online tutorial |
| **Understanding of Clients** | The ability to understand their goals, analyse their requirements and needs, and understand of software from their perspective. | Yes | Communication with the clients, Reading past developers’ documentations |
| **Project Management** | The ability to plan the project schedule and track the current processes. | Yes |  |
| **Collaboration** | The ability to build a good relationship and cooperate with team members, supervisor and clients. | Yes |  |
| **Quality Assurance** | The ability to prevent any mistakes and avoid problems during the project. | No | To be planned |
| **Risk Management** | The ability to identify, evaluate and prioritise the risks that could be faced during the project so the risks can be minimised and controlled. | No | To be planned |

## 4.3 Technical Capabilities

|  |  |  |  |
| --- | --- | --- | --- |
| Skill | Description | Presence in Team | Plan for Upskilling |
| **Python** | The back-end programming language for the existing UpStage. | Partially | Workshop |
| **GitHub** | Version and source control tool to manage the project. | Yes | Workshop |
| **HTML5** | Markup language for front-end. | Yes | Workshop |
| **JavaScript** | Programming language of HTML and the Web. | Yes | Workshop |
| **CSS** | Language that describes the look and formatting of HTML. | Yes | Workshop |
| **Trello** | This will be used as our online story board. | Yes | Workshop |

# Scope and Objectives

UpStage currently has an existing version and a testing version run by AUT local server. This project’s objective is to maintain the current UpStage, fix all of its bugs, design and develop the first version of the new UpStage based on the structure of the current UpStage. It has to be web based and open source. A critical criterion for the new UpStage is ubiquity in access which means supporting mobile devices and tablets.

Users will be able to access a web page to join any existing channels (called as “stage”) as an anonymous audience. They will only be able to communicate with people who are in the same channel as them.

If the user wants to create a channel as a host, he/she needs to log in his/her account first. The objective of create an account is to gain access to the media items which UpStage supports like audio, animation, draw, text, and so on. Each Channel allows players to interact with other players and the audience in real time.

The main focus of the new UpStage is to make it synchronized, fast, easy to use and ensure that it supports mobile technology while the team maintains a working version of the existing UpStage.

# Project Approach

## 6.1 Project Methodology

A methodology is necessary in order to ensure a project’s success. However, it has to be appropriate and suitable for the project, as different methodologies cover different aspects and problems. For this project, we decided that an Agile methodology would be suitable. After weighing our options, we decided that Kanban is the methodology that would work best for this project.

### 6.1.1 Agile

Agile is a collection of software development methodologies which follow the Agile Manifesto. Although they follow the same principles, they have different means to achieving those principles. The main focus of Agile methodologies is to be able to adapt to changes rapidly throughout the project. It also emphasizes face-to-face communication whenever possible and frequent meetings (Sewell, 2012). Most of the methodologies that fall under Agile make use of short sprints or iterations to incrementally produce potentially shippable product, making use of the ability to incorporate client and stakeholder feedback to improve the process as the product is continually developed.

### 6.1.2 Methodology

The methodology that we have decided to use for this project is a mix of various methodologies. To track the progress of the project, we will use a story board and story cards. The number of cards on each column at any point of time will be limited, following Kanban’s philosophy of limiting the Work-in-Progress to reduce multitasking and work overload. Story cards will be worked on and completed in iterations, with an iteration backlog which will contain a number of user stories from the product backlog. The user stories will be sorted according to priority, to ensure that the most important features are implemented first. The iterations will be timeboxed, following the Scrum methodology. If there are unfinished user stories at the end of an iteration, they will be put back into the product backlog, which is then re-prioritized.

### 6.1.3 Rationale

Although there are many methodologies which can be used for this project, we decided to use a mixture of different methodologies for the following reasons.

1. Due to the uncertain nature of university work load and schedule, the flexibility of Agile is necessary.
2. In the same vein, as the project team is newly formed and cannot properly gauge the effort needed for the project, the flexibility to allow changes is necessary.
3. Iterations allow the product to be produced incrementally, which will allow the project team to get feedback from the client between iterations, greatly reducing the chance of failure in delivering what the client wants.
4. Iterations allow the project team to prioritize functions or parts of the product which the client deems more important, producing value for the client at a faster rate.
5. The use of a board and cards helps visualize the project’s progress, and to notice fluctuations in the workflow speed.
6. It encourages collaboration and teamwork, allowing team members to combine their strengths and cover their weaknesses.
7. Timeboxing the iterations will reduce the chance of delays or procrastination.

### 6.1.4 Project Tasks

|  |  |
| --- | --- |
| Task | Description |
| Team Meetings | The project team will meet to discuss the tasks assigned to each team member, or if they face any problems. These meetings will be held weekly, and any time the team feels they are needed. |
| Supervisor Meetings | The project team will meet with the project supervisor to discuss the progress of the project and ask for advice on any problems faced. These meetings will be held fortnightly or when the team or supervisor feels they are needed. |
| Client Meetings | The project team will meet the client to discuss the progress of the project and ask for feedback on the work done. This is also when re-prioritization of the product backlog will happen, if it is needed. These meetings will be held at the end of each iteration or when the client feels they are needed. |

|  |  |  |
| --- | --- | --- |
| Kanban Board Column | WIP Limit | Description |
| Product Backlog |  | The remaining user stories which need to be completed. These user stories will be ordered based on their priority, and may change throughout the project. |
| Developing | 5 | User stories that are currently being worked on. |
| Testing | 3 | User stories which need to be tested, or are currently being tested. |
| Done |  | User stories which have been tested and is ready for client review. |
| Expedite |  | Urgent and sudden user stories which need to be completed before work on the other user stories is resumed. |

### 6.1.5 Project Deliverables

|  |  |
| --- | --- |
| Deliverable | Description |
| Product Backlog | A list containing all the user stories that need to be completed. |
| Project Proposal | A proposal detailing the initial plan and approach to be used for the project. |
| Mid-Project Review | A review of what has been completed and the progress of the project. |
| New UpStage | A version of UpStage with all user stories in the product backlog completed. |
| Fixes to Current UpStage | Maintenance and bug fixes of the current UpStage |
| Poster |  |
| Portfolio |  |
| Logbook | Personal logbooks of each team member as evidence of the work done. |

## 6.2 Alternative Methodologies

Amongst the Agile methodologies, several methodologies fulfill some of our needs but not all of them. However, they can possibly be adopted and integrated into the workflow as the project progresses. These are several of the likely candidates for alternative methodologies that can be used for the project.

### 6.2.1 Scrum

Scrum is an Agile methodology which makes use of a story board and story cards to track the project’s progress. Scrum utilizes short sprints, usually 2-4 weeks long, where the project team will be working on the user stories in the sprint backlog. This is then repeated until the product backlog is empty. Scrum sprints are timeboxed, so unfinished user stories will be put back into the product backlog, and may or may not be included in the next sprint, depending on re-prioritizations.

### 6.2.2 Feature Driven Development

Feature Driven Development (FDD) is a client-centric, architecture-centric and pragmatic software process (Agile Modeling, n.d.). A development cycle includes developing a model, building a feature list, assigning those features to team members, designing a sequence diagram or plan for the features which will be developed, and finally building those features. However, as FDD is designed for multiple teams to complete sets of tasks, we cannot fully adopt this methodology. We can however follow the approach of extracting important features from the product to aid in creating the product and iteration backlog.

### 6.2.3 Test Driven Development

Test Driven Development (TDD) is another Agile methodology which can be used for this project. It is helpful for making sure that the product requirements are properly understood. By making tests before coding and then coding just enough to pass those tests, it reduces the chance of wasted time on unnecessary features or code. A development cycle would be to write the tests for a specific user story or feature, then making sure that the test does not pass without the feature being implemented. The developers would then start coding the feature into the existing code, making sure that any changes made do not introduce bugs which did not previously exist. Finally, when the tests pass, the cycle is repeated, with tests for the next feature created.

## 6.3 Project Development Tools

For this project, we will need to do extensive research and evaluation to decide on the proper language or tools to use for developing the product. However, there are some options which we think stand out, although these might change as more research is done and a deeper understanding of the domain is obtained.

### 6.3.1 Front-End Options

For the frontend, there are several options which we need to evaluate. Our top two choices currently are HTML5 and JavaScript. HTML5 supports video and audio, which will remove the need for Flash should it be used to develop the new UpStage. It is also widely supported and compatible with most platforms, which is good for this project. JavaScript is a language which is commonly used with web browsers. It is supported by most web browsers and mobile devices without plug-ins, while also allowing the use of Node.js for the backend, making it a good choice as well.

### 6.3.2 Back-End Options

For the backend, we also have multiple options that need evaluation. The two options we think is appropriate are Node.js and Java. Node.js, which is written in JavaScript, can be used. Using Node.js for the backend while using JavaScript to develop the frontend has the advantage of being consistent, reducing the risk of failure. It is also a relatively simple language, which is helpful. Java is the main language taught to AuT students. Using Java as the primary language for developing the backend would mean less time spent on upskilling for new students who will join the UpStage project.

### 6.3.3 Architecture Pattern Options

For this project, the main option we will be evaluating is the MVC, or Model-View-Controller pattern. It separates the three parts of an application. The Model is the part which stores the data. The View is the part which handles displaying the information to the user. It takes the data from the Model and displays it to the user. The Controller is what tells the Model when to update its data or when the View should update its display. By separating these parts, it would reduce the risk of functions interfering with another function. Also, it would make finding the source of bugs which only affect one of the parts easier.

# Project Plan

## 7.1 Initial Project Schedule

|  |  |  |  |
| --- | --- | --- | --- |
| Project Aspect | Duration (hour) | Start Date | End Date |
| UpStage Project | 1380 | 20/07/2015 | 24/06/2016 |
| Create Project Proposal | 120 | 20/07/2015 | 12/08/2015 |
| Investigate and identify necessary technology | 100 | 13/08/2015 | 19/08/2015 |
| Upskilling | 100 |  |  |
| Requirements Confirmation | 20 | 20/08/2015 | 26/08/2015 |
| (Mid Semester Break) |  | 29/08/2015 | 13/09/2015 |
| Spike – Software Architecture and Design | 70 | 14/09/2015 | 20/09/2015 |
| Develop the New UpStage |  | 21/09/2015 | 24/06/2016 |
| Maintain the Existing System |  | 21/09/2015 | 24/06/2016 |
| Spike – build baseline infrastructure | 70 | 21/09/2015 | 27/09/2015 |
| Iteration 1 | 270 | 28/09/2015 | 18/10/2015 |
| Review with clients | 10 | 19/10/2015 | 19/10/2015 |
| Mid Project Review | 24 |  |  |
| Iteration 2 | 270 | 23/10/2015 | 12/11/2015 |
| Review with clients | 10 | 13/11/2015 | 13/11/2015 |
| (Summer Break) |  | 02/11/2015 | 29/02/2016 |
| Iteration 3 | 60 | 01/03/2016 | 21/03/2016 |
| Review with clients | 4 | 22/03/2016 | 22/03/2016 |
| Iteration 4 | 60 | 23/03/2016 | 12/04/2016 |
| Review with clients | 4 | 13/04/2016 | 13/04/2016 |
| (Mid Semester Break) |  | 11/04/2016 | 24/04/2016 |
| Iteration 5 | 60 | 25/04/2016 | 15/05/2016 |
| Review with clients | 4 | 16/05/2016 | 16/05/2016 |
| Iteration 6 | 60 | 17/05/2016 | 06/06/2016 |
| Review with clients | 4 | 07/06/2016 | 07/06/2016 |
| Poster presentation | 20 | 08/06/2016 | 14/06/2016 |
| Final Report | 40 | 15/06/2016 | 24/06/2016 |

\*Iteration: Team will do tasks from product backlog. (Please see appendix)

## 7.2 Quality Assurance

### 7.2.1 Quality Assurance Plan

To ensure that the quality of the product produced is acceptable, we have created a list of activities which need to be done before each client review meeting.

#### 7.2.1.1 Requirements Review

* The client must agree with the product backlog produced by the team
* The backlog must be ordered according to priority with the help of the client
* The team must understand the project’s requirements and the scope of the project

#### 7.2.1.2 Coding

* The code produced must follow programming standards
* The code must be appropriately commented
* The code must be appropriately documented
* The code must pass all the tests
* Pair programming will be utilized when applicable to increase the quality of code produced and reduce errors
* Code that is produced without Pair Programming will be reviewed by another team member

#### 7.2.1.3 Testing

* The code must be thoroughly tested at the end of each iteration
* Unit tests for the code must be written and passed
* Regression tests for the code must be written and passed
* Test results must be recorded
* A defect log must be maintained throughout the project

#### 7.2.1.4 Deliverables

* The deliverables must be updated and revised throughout the duration of the project
* The documents must be readable and written in a professional manner

#### 7.2.1.5 Supervisor and Client Meetings

* The project team must prepare a meeting agenda before each meeting
* The project team must record meeting minutes and send them to the supervisor or client and confirm that there are no misunderstandings

#### 7.2.1.6 Logbooks

* Each team member must maintain a logbook or blog as evidence of their work

### 7.2.2 Quality Assurance Tools

|  |  |
| --- | --- |
| QA Tool | Function |
| Github | A web-based version control and collaboration tool, it allows the team to share code and files. It ensures that the team is notified when a new version of a file(s) is uploaded, thus reducing the risk of working on obsolete versions. |
| Trello | Trello is the Project Management Software that we will use for this project. It is a web-based PMS which is designed to work with the KANBAN methodology, simulating a physical board and cards. It will be used to track the progress of the project. However, a physical board will also be maintained. Screenshots of the virtual board will be periodically taken as evidence of the workflow. |

## 7.3 Review Plan

* The project schedule will be revised after each iteration
* The risk register will be revised and updated when new risks are found
* The product backlog will be revised and updated when new requirements are added or an existing requirement is changed
* The Quality Assurance Plan will be revised and updated if the current plan is found to be ineffective

# 7.4 Estimate all costs incurred

## 7.4.1Student Costs

Ongoing Team Members:

Siatua Uili: 240 hours for the 2nd semester in 2015

William Stokes: 240 hours for the 2nd semester in 2015

New Team Members:

Alyssa Byun: 300 hours for the 2nd semester in 2015

Jing Han: 150 hours for the 2nd semester in 2015  
 and 150 hours for the 1st semester in 2016

Joshua Kartono: 150 hours for the 2nd semester in 2015  
 and 150 hours for the 1st semester in 2016

Total:

Total hours for the 2nd semester in 2015 are 1080 hours.

Total hours for the 1st semester in 2016 are 300 hours. (This excludes the hours of any new team members joining in 2016.)

Therefore, the total hours to be spent on UpStage project over the year are 1380 hours.

## 7.4.2 University Costs

* Supervisor’s time spent for meetings with the team
* Workstations for the team
* Internet access
* Computer lab access
* Software for the project (Microsoft Office, etc.)
* Printer/Photo copier

## 7.4.3 Client Costs

* Time spent for any communication including meetings and emails with the development team.

## 7.5 Communication Plan

* Kick off meeting

-Ensure each person’s role.

-Get familiar with the existing project.

-Clear statement about what we are going to deliver to the sponsors.

* General meeting

-General meeting will take place at least once per week.

-Check the current progress.

-If team members face any issues, we will attempt to solve it together during the meeting.

-Make a plan for the following week’s tasks.

-Complete the meeting minutes.

* Supervisor meeting

-Supervisor meeting will take place once per week or fortnightly.

-Review the project.

-Make sure the project is in the right track.

-Complete the meeting minutes.

* Client meeting

-Client meeting will take place at the end of each iteration.

-Make sure that the client is satisfied with the current progress.

-Complete the meeting minutes.

* Team communication

-Team members communicate through Facebook.

-Logs of communication will be recorded.

-Any updates done should be posted on Facebook and notified to clients and supervisor through E-mail.

## 7.5.1 Method for updating the communications management plan

Any advice or request from any other stakeholder must be made through the project team and approved/disapproved by the project manager with final approval residing with the project sponsor.

## 7.5.2 Escalation procedures

|  |  |  |  |
| --- | --- | --- | --- |
| **Priority** | **Definition** | **Decision Authority** | **Timeframe for Resolution** |
| 1 | Great impact to project that may cause complete failure of the project if not fixed quickly. | Client and Supervisor | Within 2 hours |
| 2 | Great impact to project that may cause partial failure to meet project deadlines, cost frames or key project requirements. | Client and Supervisor | Within 4 hours |
| 3 | Medium impact on the project that may cause conflicts to meet deadlines, cost frame or quality frame. | Project Leader | Within 1 meeting day |
| 4 | Insignificant impact to project that needs to be resolved to ensure project is on track | Project Leader | Within 2 meeting days |

## 7.6 Risk Management Plan

1 – Minor 2 – Moderate 3 – Major

Low – Unlikely Medium – Possible High – Almost certain

P – Probability I – Impact Score

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| # | Risk | Cause | Category | P | I | Impact | Mitigation | Risk owner |
| 1 | Fail proposal | Proposal is rejected by panel | Assessment | Low | 3 | Wasted time on resubmission, or worse repeating the paper. | Read project proposal requirements carefully | Team |
| 2 | Team member leaves | Student stops taking paper | Team management | High | 2 | Task must be reassigned, less man hours to complete the work. | Have a cross functional team that codes in pairs | Team |
| 3 | Sickness | Team member gets sick | Performance | High | 2 | Cannot attend meetings, and possibly not do work. | Keep healthy | Team |
| 4 | Poorly identified requirements | Failed to properly understand the clients requirements | Requirements | Low | 3 | Unable to meet the client’s needs. | Identify and understand requirements and send a copy of requirements to the client to double check | Team |
| 5 | Client changes requirement | Client has new requirements | Requirements | Medium | 2 | More development needed, possibly redevelop completed tasks  Wasted time | Get approval on tasks from client before development starts | Client |
| 6 | Conflict between team members | Disagreement between team members | Team/ Cooperation | Low | 1 | Decision cannot be settled | Have healthy constructive discussions about disagreements | Team |
| 7 | Conflict between team and client | Disagreement between team members and client | Team/ Cooperation | Low | 2 | Loss of cooperation and wasted time | Have healthy constructive discussions about disagreements | Team |
| 8 | Conflict between team & supervisor | Disagreement between team members and supervisor | Team/ Cooperation | Low | 2 | Loss of cooperation | Have healthy constructive discussions about disagreements | Team |
| 9 | Team members do not cooperate with each other | Disagreement between team members | Cooperation | Low | 3 | Less work completed, less productivity | Have healthy constructive discussions about disagreements | Team |
| 10 | Loss of work | Latest work that was created is lost | Technology | Low | 3 | Wasted time and effort redoing what was already done | Keep an up to date backup of work | Team |
| 11 | GITHUB goes down | Cannot access GitHub to obtain the latest version of the project | Technology | Low | 3 | Wasted time not able to work and possibly rebuilding tasks | Keep an up to date backup of work | Team |
| 12 | Task estimates are off | Underestimated or overestimated tasks | Performance | Medium | 2 | Less time for other features | Assess previous work and base future work off of the requirements | Team |
| 13 | Trello goes offline | The server for the project management tool goes down | Technology | Low | 3 | Task documentation is inaccessible | Keep an up to date backup of work | Team |
| 14 | Failure to turn up to meetings | Team member(s) fail to turn up to meetings | Team management/ Cooperation | Low | 2 | Team has to go over with the member(s) what was covered, wasted time |  | Team |
| 15 | Team member is unavailable for long periods of time | Student goes on holiday, personal problems | Team management/ Cooperation | Low | 2 | Delays in the work delivered, less work delivered | Keep mobile on you at all times. Always check your emails, text, Face book or Trello | Team |
| 16 | Chosen Programming language hard to achieve tasks | Not enough research before developing | Technology | Low | 3 | Team has to change other suitable programming language. Wasted time | Do enough research for the programming language and choose carefully | Team |
| 17 | Cannot fix existing bugs | Not familiar with chose programming language | Technology | Low | 2 | Wasted time | Skill up, read previous relevant report | Team |

## 

## 7.7 Definition of Done

The project is done when:

* All user stories in the product backlog are completed
* Code is properly commented and refactored
* Code is tested and no errors come up
* Final product build passes all tests written for it
* Product runs without errors on agreed platforms
* Any changes to the plan or documents are documented
* The clients are satisfied with the final product
* All deliverables are produced

# 

# Disclaimer

**Auckland University of Technology**

**Bachelor of Computer & Information Sciences**

**Research & Development Project**

**Disclaimer:**

**Clients should note the general basis upon which the Auckland University of Technology undertakes its student projects on behalf of external sponsors:**

While all due care and diligence will be expected to be taken by the students, (acting in software development, research or other IT professional capacities), and the Auckland University of Technology, and student efforts will be supervised by experienced AUT lecturers, it must be recognised that these projects are undertaken in the course of student instruction. There is therefore no guarantee that students will succeed in their efforts.

This inherently means that the client assumes a degree of risk. This is part of an arrangement, which is intended to be of mutual benefit. On completion of the project it is hoped that the client will receive a professionally documented and soundly constructed working software application, some part thereof, or other appropriate set of IT artefacts, while the students are exposed to live external environments and problems, in a realistic project and customer context.

In consequence of the above, the students, acting in their assigned professional capacities and the Auckland University of Technology, disclaim responsibility and offer no warranty in respect of the “technology solution” or services delivered, (e.g. a “software application” and its associated documentation),both in relation to their use and results from their use.

# Glossary

|  |  |
| --- | --- |
| Work | Meaning |
| Python | A widely used general purpose, high-level programming language |
| CSS | a style sheet language used for describing the look and formatting of a document written in a markup language. |
| GitHub | Online distributed revision control and source code management (SCM). |
| HTML5 | A core technology markup language of the Internet used for structuring and presenting content for the World Wide Web. |
| JavaScript | Language used to make web pages interactive |
| Trello | A free web-based project management application |
| MVC | Model view controller |
| Flash | A multimedia and software platform used for creating vector graphics, animation, browser games, rich Internet applications, desktop applications, mobile applications and mobile games. |
| Node.js | an open source, cross-platform runtime environment for server-side and networking applications. |
| Timebox | A fixed time period which is allocated to an activity. |

# Appendix

## 10.1 Product Backlog

**Audience**

|  |  |
| --- | --- |
| Backlog | Priority |
| Can access a link to find user’s local time in the homepage | NICE TO HAVE |
| Can enter a stage | MUST HAVE |
| Can send text messages in the stage | MUST HAVE |
| Can register to be a Player | MUST HAVE |
| Sign in with the registered account | MUST HAVE |

**Player**

|  |  |
| --- | --- |
| Backlog | Priority |
| Has the same functionalities as audience |  |
| Can log out | MUST HAVE |
| Will have tool bar in the stage if the player is assigned to this stage | MUST HAVE |
| Can put avatars in the stage | MUST HAVE |
| Can move the avatars in the stage | MUST HAVE |
| Can stop the avatars in the stage | MUST HAVE |
| Can select a prop for the avatar | MUST HAVE |
| Can select a background | MUST HAVE |
| Can clear the avatars in the stage | MUST HAVE |
| Can play audio in the stage | MUST HAVE |
| Can turn volume up/down for audio in the stage | MUST HAVE |
| Can stop a single audio | MUST HAVE |
| Can stop all audio | NICE TO HAVE |
| Can rename their current avatar | MUST HAVE |
| Can draw line with chosen colour in the stage | NICE TO HAVE |
| Can select the type of line to draw | NICE TO HAVE |
| Can change the thickness of line to draw | NICE TO HAVE |
| Can clear the line | NICE TO HAVE |

**Creator**

|  |  |
| --- | --- |
| Backlog | Priority |
| Can log out | MUST HAVE |
| Create a new Stage | MUST HAVE |
| Enter the stage’s full name and short name as a URL | NICE TO HAVE |
| Edit a stage | MUST HAVE |
| Modify the stage’s full name and short name for URL | NICE TO HAVE |
| Modify the splash message | NICE TO HAVE |
| Can choose if the debug messages are printed on the backdrop |  |
| Can change the props and backdrops’ colour | NICE TO HAVE |
| Can change the chat window’s colour | NICE TO HAVE |
| Upload media so that it can be used in stage | MUST HAVE |
| Choose the type of media for uploading: avatar, prop, backdrop, audio or video-avatar | MUST HAVE |
| Modify media’s name and tags | NICE TO HAVE |
| For avatar, a voice and be chosen | MUST HAVE |
| The chosen voice can be tested | MUST HAVE |
| The number of frames for animated avatars can be selected | NICE TO HAVE |
| A local file can be chosen to attach | MUST HAVE |
| Choose the stages to assign the media to | MUST HAVE |
| For prop, the number of frames can be selected | NICE TO HAVE |
| For backdrop, the number of frames can be selected | NICE TO HAVE |
| For audio, choose the type of this audio: sound effect or music | MUST HAVE |
| For video-avatar, choose from existing streams in the /media/video directory | MUST HAVE |
| View an uploaded media | MUST HAVE |
| Display specific media by choosing filters | NICE TO HAVE |
| Add more filter to display an eligible media | NICE TO HAVE |
| Can remove the filter | NICE TO HAVE |
| Can reset the searching filter and result | NICE TO HAVE |
| Search media | MUST HAVE |
| Edit an uploaded media by modify name, voice and the stages to assign | MUST HAVE |
| Can display the detail of the media | MUST HAVE |
| Can save changes | MUST HAVE |
| Can delete the media | MUST HAVE |
| Delete the media even if in use | NICE TO HAVE |
| Edit players | MUST HAVE |
| Can modify their password | MUST HAVE |
| Can modify their email | NICE TO HAVE |
| Can modify their user type | MUST HAVE |
| Enter a stage | MUST HAVE |
| Can send text message in the stage | MUST HAVE |
| Be assigned to a stage | MUST HAVE |
| The message sent can be read out as a chosen voice | MUST HAVE |
| Has the tool bar (same toolbar as player’s) in the stage if the creator is assigned to this stage | MUST HAVE |
| Can create a new player account | MUST HAVE |
| Can enter the username, password, confirm password and email address | MUST HAVE |
| Select the type of the new player account: player, maker, unlimited maker, admin or creator | MUST HAVE |
| Be able to save | MUST HAVE |
| Can edit existing player details | MUST HAVE |
| Can display the detail of all existing player accounts | MUST HAVE |
| Can select one of them to change password | MUST HAVE |
| Update the changes | MUST HAVE |
| Delete the player’s account | MUST HAVE |
| A link to edit page mode | MUST HAVE |
| Can edit the homepage | MUST HAVE |
| Can edit the workshop | MUST HAVE |
| Can edit the player page | MUST HAVE |
| Can edit the stages page | MUST HAVE |
| Can edit the sign up page | MUST HAVE |
| Submit the changes | MUST HAVE |
| Reset the page to be default | NICE TO HAVE |

**Maker**

|  |  |
| --- | --- |
| Backlog | Priority |
| Has the same functionality as creator except they can’t create new players or edit | MUST HAVE |
| Has an upload limit for file sizes when uploading files (10MB) |  |

**Admin**

|  |  |
| --- | --- |
| Backlog | Priority |
| Has the same functionality as creator but can’t edit or create creators | MUST HAVE |
| Has an upload limit for file sizes when uploading files |  |

## 10.2 Team Contract

As a team member, I will:

* Show respect for myself and other members
* Always try to do my best work
* Be kind and helpful to other members
* Listen to each other’s ideas with respect
* Work toward the understood goals of the team
* Be enthusiastic about working with the team
* Follow through on commitment
* Take pride in the team’s work
* Readily accept feedback on performance
* Encourage others to achieve at high levels
* Be able to stay focused on team tasks
* Openly communicate with others in the team
* Be able to resolve conflict effectively
* Get to know the other members
* Not be late for the meeting without proper reason
* Attend every meeting unless there is a proper reason

If someone on our team break one or more our rules, the person will be asked to follow our agreement. If the person still breaks the rules, we will ask our supervisor to help find a solution.

Date: 20/08/2015

Team Member Signature:

|  |  |
| --- | --- |
| Alyssa Byun | C:\Users\wbv5134\Desktop\Alyssa.PNG |
| Jing Han | C:\Users\wbv5134\Desktop\jing.PNG |
| Joshua Kartono | C:\Users\wbv5134\Desktop\joshua.PNG |
| Sia Uili | C:\Users\fdr2562\Desktop\11907989_10205061366717771_1051273972_n.jpg |
| William Stokes | C:\Users\wbv5134\Desktop\william.PNG |

# References

*Agile Modeling (n.d.). Feature Driven Development (FDD) and Agile Modeling.*

*Retrieved from: http://www.agilemodeling.com/essays/fdd.htm*

*Sewell, L. (2012). Agile Software Development. Delhi: Research World.*

*UpStage (n.d.) Retrieved from UpStage: http://upstage.org.nz/*

# 