Contents

1 Development Framework

The team has decided to adopt a **Scrum framework** in our **Agile approach** to deliver this software.

2 Sprints

The team will adopt multiple sprints throughout the duration of the project, whose dates and goals are defined by a group decision and with the client.

2.1 Sprint Dates & Goals

The following lists each of the Sprints, their dates and intended goals. This list will be updated as new sprints are devised throughout the development lifecycle.

2.1.1 Semester 1

1. **March 7 to March 21** - This sprint was decided in the March 7 meeting[1] with our client and is intended to "consist of small bug fixes or enhancements that will help the team familiarise themselves with Doubtfire's codebase"
2. **March 21 to April 4** - This sprint builds upon the first sprint whilst concurrently working on requirements documents and assessment criteria for the project. Jake and Reuben will take on unit testing as it has been neglected in the past for the Rails API and will investigate methods to test it.
3. **April 4 to May 4** - This sprint is the largest sprint of the semester. It focuses on the team dedicating time to shift their resources on a majority of requirements analysis for the system, including prototyping and architecture decisions. Like Sprint 2, it is expected that team members also work on tasks as need be for practise and familiarisation of the Doubtfire codebase
4. **May 4 to May 23** - This sprint focuses on the film which needs to be shot by the presentation on May 23 in Week 11. In addition, this sprint will focus on the majority of the requirements documentation that needs to be properly thought about, such as use case descriptions.

[1] See `https://github.com/final-year-project/documentation/wiki/Meeting-Minutes#re-4-spiking-doubtfire-for-the-next-few-weeks`

5. **May 23 to May 30** - This last sprint for Semester 1 wraps up the first semester as the team prepares their portfolio for submission and assessment. It should include a thorough review of the documents produced thus far, and make additional 'last-minute' changes if need be.

2.1.2 Midyear Break

0. **May 30 to August 1** - This sprint, over the midyear break, aims to develop the core API codebase changes needed to introduce the helpdesk ticketing system. It is aliased to Semester 2's 'zero'th' sprint.

2.1.3 Semester 2

1. **August 1 to August 22** - This first Semester 2 sprint aims on locking the API code. All testing for the API should be done by this stage. Begin planning the UI changes needed such as modals to create tickets and sessions which tutors can clock on/clock off from the programming helpdesk.
2. **August 22 to September 5** - This sprint aims at having the Dashboard UI planned in addition to beginning user manual documentation on how to create and submit tickets as well as implementing some partial functionality of the dashboard.
3. **September 5 to September 26** - This sprint aims at completing as much usability testing as possible on the dashboard, being the most integral part of the software.
4. **September 26 to October 10** - This sprint focuses on getting the dashboard finalised and adding in as many performance improvements to the UI as possible. The team also will focus on the presentation and getting it finalised in this time.
5. **October 10 to October 31** - This sprint will focus on finalising the codebase and resolving all conflicts between the origin Doubtfire codebase and the new Doubtfire codebase with new features added.
6. **October 31 to November 6** - This sprint will focus on the team finalising the portfolio for printing and submission.

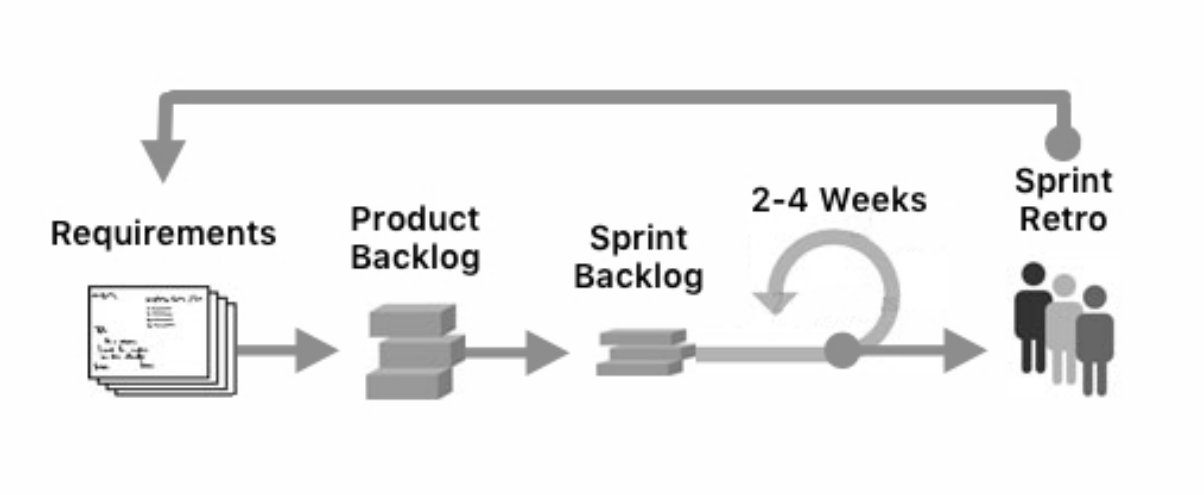

Figure 1: High level overview of the workflow

3 Development workflow

3.1 High level overview

The high-level overview of the workflow is an adapted standard scrum process that better suits the needs of the project:

1. The **requirements** of the project is devised with the client in **ongoing client meetings**. This means that requirements can be added on an ongoing basis throughout the project
2. Requirements are fed into the **project backlog**, that is the backlog of the entire Doubtfire system, as tasks
3. Specific tasks are chosen to be worked on a sprint into a **sprint backlog**
4. The sprint lasts for 2-4 weeks, where tasks from the sprint backlog are eventually all completed
5. A **sprint retrospective** is run amongst both the team and the client, where feedback from the sprint is eventually fed back into initial the requirements

There are no daily stand-ups like the standard scrum framework mandates. This is due to time and constraints with the group, as only one or two meetings are possible throughout the week. The team aims for a meeting together once a week, and a meeting with the client on a fortnightly basis.

To slimline the workflow, the sprint review process is merged into the sprint retrospective, meaning the retrospective is conducted with the team and client such that the client gets a feeling of how the team is progressing through tasks *and* can suggest changes to the product in one go.

The workflow enhances Doubtfire's **continuous integration**[2] as each changes will be pushed to the `develop` branch, and eventually `master` branch (meaning that as tasks get completed, the code changes will go live to production).

3.2 Trello Boards

As described by the Project Tools[3] document, Trello is being used to manage tasks. The Trello Workflow adapts to the high level overview in the form of variant boards and columns. Each board and its intended workflow is outlined below in further detail.

3.2.1 Doubtfire Backlog Board

The **Doubtfire Backlog Board**[4] is the *Product Backlog* for Doubtfire. It aims to outline all tasks in Doubtfire's backlog horizontally. Each column is described as thus:

1. **Fresh Ideas** - New tasks that the product owner thinks of will be added here. But those tasks should be be moved into one of the other lists on this board as soon as possible, and therefore this list should be kept as empty as possible.
2. **Quick and Easy**
3. **Top Priority** - Tasks that need to be completed ASAP, such as critical bugs
4. **High Priority** - Tasks that need to be completed which have high importance, such as bugs
5. **Medium Priority** - Tasks that should be completed soon, such as new features or enhancements
6. **Low Priority** - Tasks that would be nice to have done eventually, such as small UI beautifications
7. **One Day** - Ideas that would be great to have *one day* if we had the time, essentially tasks that are currently too out of scope

[2] See `https://en.wikipedia.org/wiki/Continuous_integration`

[3] See `https://github.com/final-year-project/documentation/wiki/Project-Tools#trello-task-management`

[4] See `https://trello.com/b/0uh6AZdu/doubtfire-backlog`

Figure 2: The Doubtfire Backlog Board

8. **Maybe...** - Ideas that are still being considered

The Doubtfire Backlog Board is maintained by the Product Owner (i.e., the client)—the team does not have access to add new tasks here unless it is approved by the Product Owner.

3.2.2 Helpdesk Ticketing System Board

The **Helpdesk Ticketing System Board**[5] is the board that outlines all tasks for the team's project.

This board not only contains the **Sprint Backlog** tasks, but also meta tasks related to administration and assessment that needs to be done (such as organising meetings with clients, installing required software etc). This will help the project manager organise which tasks the team needs to not only on a *developmental* basis, but also on an *assessment* and *administration* basis for the final year project unit.

The board is organised into several columns:

[5]See `https://trello.com/b/8a1k0Wud/helpdesk-ticketing-system`

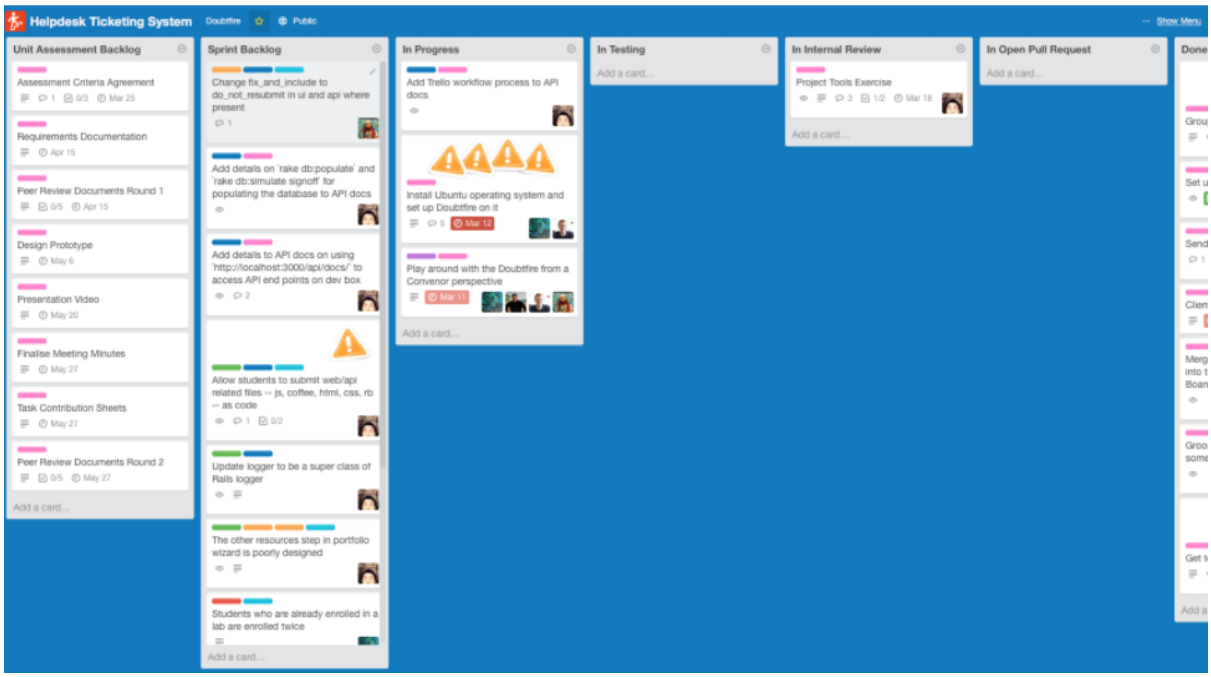

Figure 3: The Helpdesk Ticketing System Board

1. **Unit Assessment Backlog** - The backlog of tasks that are required by the final year project unit deliverables[6]
2. **Sprint Backlog** - Tasks involved over the upcoming time-fixed sprint moved from the Doubtfire Backlog board - essentially the current sprint backlog
3. **In Progress** - Tasks that are currently in progress. These tasks **must be assigned to whoever is working on them**
4. **In Testing** - Where relevant, tasks move into this column if unit or integration testing is needed on that task (e.g., a new API endpoint should have unit tests written). **These tasks should be assigned to whoever is writing the tests for the task**
5. **In Internal Review** - Tasks that are to be reviewed internally by another team member. **These tasks should be reassigned to the reviewer**
6. **In Open Pull Request** - Tasks that have been put into a Pull Request and assigned to a product owner for external code review. **These tasks should be reassigned to the external code reviewer**.

[6]See `https://github.com/final-year-project/documentation/wiki/Deliverables`

7. **Done** - When the task is merged into the `develop` branch (the Pull Request has been closed).

This board is maintained by the project manager of the team, and also updated by team members as they progress through their allocated tasks.

3.2.3 Trello Workflow in a Sprint

This workflow is used to guide team members on how to use Trello for their day-to-day activities whilst working on Doubtfire.

The workflow also complements the Doubtfire Git Workflow[7], meaning that all changes made will be pushed into the primary `develop` branch on the Doubtfire product workflow for improved continuous integration.

3.2.3.1 Prepare cards for a sprint

In this step, team members will go to the Doubtfire Backlog Board[8] and find (or create with the Product Owner's permission) tasks that are relevant for the current sprint.

Once they have found a card, they can move it directly to the Helpdesk Ticketing System Board[9]'s Sprint Backlog column. To do this, they can click on a card and them move it:

3.2.3.2 Progressing the card

Once the card is in the backlog on the Helpdesk Ticketing System Board, it will be progressed throughout the board (moved toward to the right hand side of the board) to describe its process by dragging and dropping it between columns:

3.2.3.2.1 In Progress

When the card is ready to start on, progress the card from the **Sprint Backlog** to the **In Progress** column.

It needs to be assigned to whoever is working on the card. To do so, click on the card and assign the task to a team member:

[7] See `https://github.com/doubtfire-lms/doubtfire-api/blob/develop/CONTRIBUTING.md`
[8] See `https://trello.com/b/0uh6AZdu/doubtfire-backlog`
[9] See `https://trello.com/b/8a1k0Wud/helpdesk-ticketing-system`

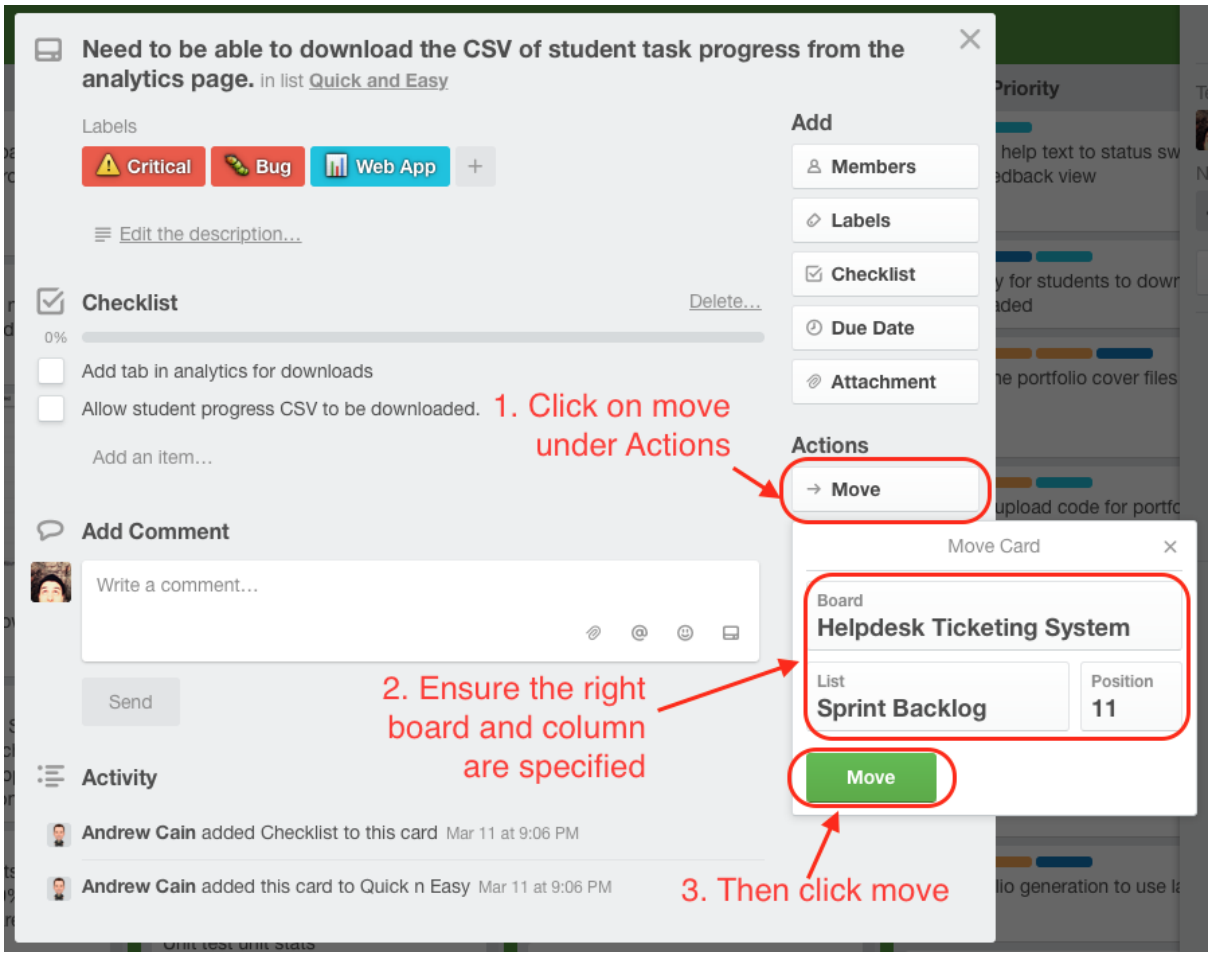

Figure 4: Moving a card to the right board

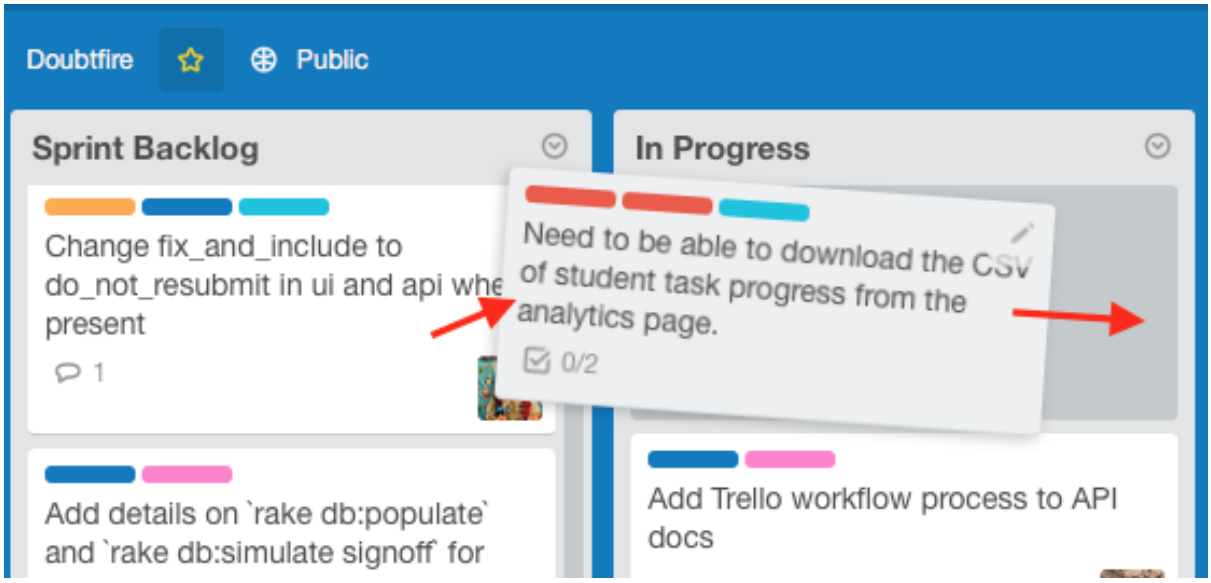

Figure 5: Dragging and dropping the card

3.2.3.2.2 In Testing

If the card requires testing to be done, progress the task from **In Progress** to **In Testing**.

3.2.3.2.3 In Internal Review

When the person working on the card thinks the card is ready, they should run a **code walkthrough** with someone else in the team.

Progress the task from **In Testing** or **In Progress** to **In Internal Review** and sit down together with the team member. Walk the other team member through both the *functionality* changes added, as well as the *code* that has been added. This ensures for optimal product quality and code quality by having a second set of eyes look over the code.

When the code review is over, remove the other team member from the members list of the card.

3.2.3.2.4 In Open Pull Request

As through the Doubtfire Git Workflow[10] a Pull Request should be submitted to the

[10]See `https://github.com/doubtfire-lms/doubtfire-api/blob/develop/CONTRIBUTING.md#`

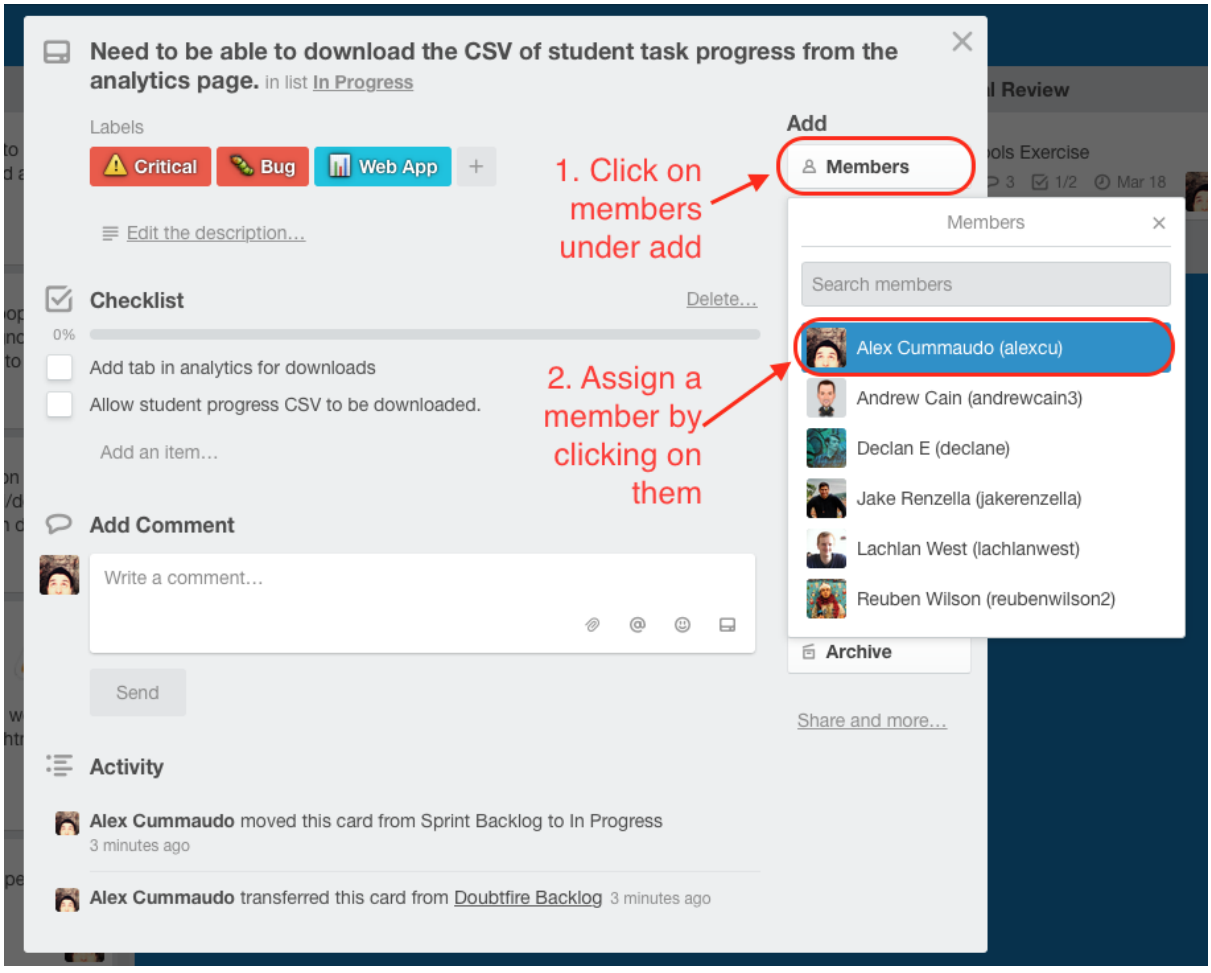

Figure 6: Assign a member

Product Owners (i.e., the `doubtfire-lms` repository) when ready for external review.

Progress the card to **In Open Pull Request** from **In Internal Review** when the pull request has been submitted and assign the Product Owner (e.g., Andrew Cain) to the card (if applicable).

3.2.3.2.5 Done

Progress the card from **In Open Pull Request** to **Done** only once the changes made from the card have been merged and the Pull Request is closed. The assigned task member can be removed from the card at this point.

`4-submitting-a-pull-request-pr-to-the-upstream-repository`