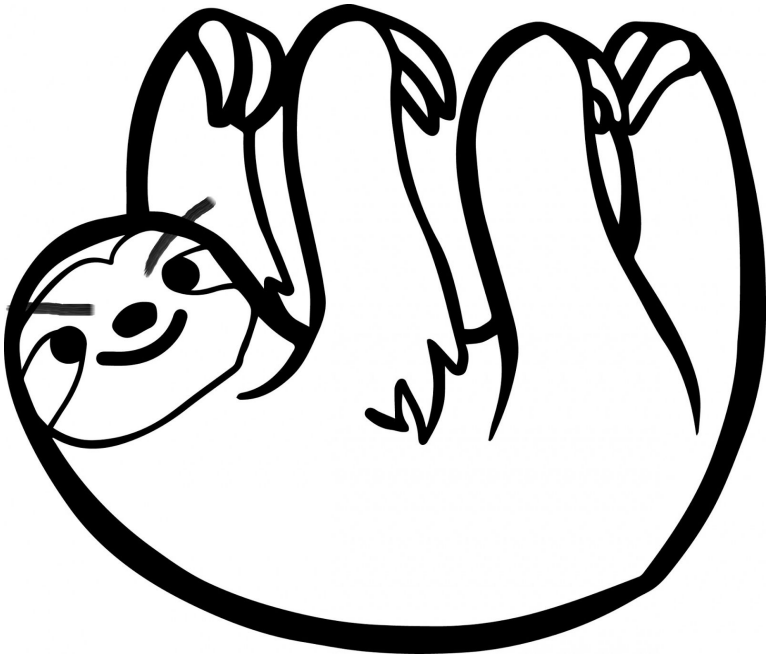
Not Fast, Just Furious



Alexander Hung, David Yau, Gavin Zhang, Jason Fong, Mathu Manogaran

Product Backlog v4

Last updated: November 20, 2017

Table of Contents

Changelog

Version 0 (Oct 15) - deliverable 2

- Initial release, contains personas, user stories, sorted user stories (by category)
- Has 4 personas: Elitist, Diligent, Average, Inattentive students

Version 0.1 (Oct 19) - deliverable 2

- Added back missing persona for Professor

Version 1 (Oct 20) - deliverable 3

- Merged the 4 student personas into 1 student
- Added a page of user stories sorted by order of development
- Removed unneeded persona points
- Removed unneeded user stories

Version 1.1 (Oct 23) - deliverable 3

- Added the changelog
- Broke down upcoming user stories down into tasks, along with story points

Version 2 (Oct 30)

- No changes

Version 3 (Nov 7)

- Added non-user story tasks (NST)
- Added U8

Version 4 (Nov 14)

- Added additional non-user story tasks (NST)

Personas

Sohee Kang - Professor (The Client)

- Female, middle age
- Cares deeply about her students, dedicated to her work and maintains a professional attitude whenever required
- Strong willed but can be convinced to compromise
- While she is dependant on technology she has limited experience with more advanced things and does not have the time to learn
- Has high expectations of technology's ability to solve her problems and sees technology as a potential solution to her problems rather than a problem in itself
- Prefers simple, fast interfaces but would also like to have options that provide more control over the program
- Willing to compromise on non-essential features, in order to get a working product on time.
- Will not be disappointed by a less than ideal program on the condition that it does work to an acceptable level of quality
- Understands the benefits of technology but prefers for it to be as simple as possible to the user

Summary: Stubborn at times but is willing to compromise on some things. Willing to have someone uses new technologies if she sees a significant benefit in it.

Diligent Student : Elizabeth

- Redoes assignments/finds outside questions in order to improve herself
- Wants solutions to be posted/viewable after deadline to compare
- Willing to help others should they come to her for help
- Willing to try new study methods to improve learning but prefers pen and paper over technology
- Prefers to work on paper, old fashioned
- Always on top of their studies
- Minor in statistics

User Stories

P: I want to be able to create problems without LaTEX or PERL
P: I want to be able to create sets of problems and upload them
P: I want to be able to edit an assignment
P: I want to be able to set a deadline for assignments
P: I want to make assignments visible to students
P: I want the assignment to be individualized
P: I want students to be able to receive the assignments
P: I want students to be able to submit their completed assignments
P: I want grades to be computed and uploaded automatically
P: I want to be able to add/remove students from the class
P: I want only students who are in the class to be able to access the system

S: I want to be able to view the class average, to compare my own grades to the class
S: I want to receive bonus marks for finishing an assignment early

S: I want to see the correct answers after the deadline has passed so I can compare them and see what I did wrong so that I can improve for next time
S: I want to be able to retry my assignments to get higher grades

S: I want to be able to see my mark as soon as possible so I don't need to go check it later
S: I want the assignment to be as easy to access as possible

Sorted User Stories by category

Assignment Setup
P: I want to be able to create simple problems that use addition/subtraction without LaTEX or PERL
P: I want to be able to create sets of problems and upload them
P: I want to be able to (delete/edit) an assignment
P: I want to make assignments visible to students
P: I want to be able to set a deadline for assignments

Accessing Assignments
P: I want students to be able to receive the assignments
P: I want students to be able to submit their assignments

Grades
P: I want grades to be computed and uploaded automatically
S: I want to be able to see my mark as soon as possible so I don't need to go check it later
S: I want to be able to view the class average, to compare my own grades to the class

Retrying Assignment
P: I want students to be able to retry assignments
S: I want to be able to retry my assignments to get higher grades

Extras
P: I want the assignment to be individualized
P: I want to be able to create complex problems without LaTEX or PERL where I can use mathematical symbols such as summation.

Class Management
P: I want to be able to add/remove students from the class

Sorted User Stories by <u>order of development</u>

These user stories will be split into sprints on next page

P - indicates professor

S - indicates student

<u>Initial setup (what needs to get done first)</u>

P: I want to be able to create simple problems that use addition/subtraction without LaTEX or PERL

P: I want to be able to create sets of problems and upload them

<u>Next steps</u>

P: I want to be able to add students to the class

P: I want students to be able to receive the assignments

P: I want students to be able to submit their assignments

<u>Future sprints</u>

P: I want to be able to set a deadline for assignments

S: I want to be able to see my mark as soon as possible so I don't need to go check it later

<u>Future sprints → "starting to do some extras"</u>

P: I want the assignment to be individualized

P: I want grades to be computed and uploaded automatically

P: I want to be able to maintain (delete/edit) an assignment

P: I want students to be able to retry assignments

S: I want to be able to retry my assignments to get higher grades

P: I want to make assignments visible to specific students

P: I want to be able to create complex problems without LaTEX or PERL where I can use mathematical symbols such as summation.

S: I want to be able to view the class average, to compare my own grades to the class complete b or in progress, and should be merged in the next sprint.

User Stories broken into tasks

Format:
(U1) P: (8)
means User Story 1; Professor; 8 story points

T1: (3)
means Task 1; 3 story points

(U1) P: (13) I want to be able to create simple problems that use addition/subtraction without LaTEX or PERL

- T1: (1) Create a window where data can be entered
  - **Is a dependency for everything GUI related**
  - *Uses JavaFX library*
- T2: (8) Save data in some format
  - *Design ways to store in database or objects (ex. ER diagram, UML diagram)*
  - *Will be discuss more in depth when we regroup*
- T3: (1) Output that data back onto the screen
- T4: (1) Be able to give input
- T5: (1) Check if it matches with answer
  - *Temporary simple input validation*
  - *Will not need to be with simple problems right away.*
  - *Just check if given input matches stored input*
- T6: (1) Output result

(U2) P: (8) I want to be able to create sets of problems and upload them

- T1: (2) Be able to upload individual questions through the GUI
  - *Question and Answer inputs*
- T2: (2) Be able to upload ASSIGNMENT files through GUI
- T3: (2) Create Object to store Questions
  - *With Question and Answer variables*
- T4: (2) Parse CSV file and store it
  - *“Question”, “Answer”*
- T5: (2) Store input properly into database
- T6: (2) Question id should automatically increment
  - **Low Priority**
  - **Dependent on U4 T1**
- T7: (4) Take a ResultSet (JDBC object) and read its information into a Question object

(U3) P: (8) I want to be able to add students to the class

- T1: (2) Be able to upload individual students through GUI
  - *StudentNumber, FirstName, and LastName inputs*

T2: (2) Be able to upload STUDENT files through GUI
T3: (2) Create Object to store Students
With StudentNumber, FirstName, and LastName variables
T4: (2) Parse CSV file and store it
StudentNumber, FirstName, LastName
T5: (2) Store input properly into database
T6: (4) Take a ResultSet (JDBC object) and read its information into a Student object

(U4) P: I want to be able to create assignments with a name and deadline
T1: (8) Set up Assignment table with all necessary fields including possible future fields
T2: (8) Set up Question table that stores all question fields and is connected properly to section and assignment
Question(SEC, AID, QID, question fields…)
T3: (4) Create GUI screen to create assignment
Should connect to main professor page
Should lead to Question page once assignment is created
Should have assignment #, name, deadline, and total question #
T4: (2) Combine elements of front end and back end together
Make it so assignment and student can write to the database
Dependent on U4 T1
(U5) P: (4) I want students to be able to receive the assignments
Dependent on U3
T1: (8) Set up student tables with all necessary fields
T2: (4) Create basic assignment page for student
Create new Scene for this
T3: (1) Create function to randomly choose k of n questions from assignment
T4: (3) Add n number of questions to the assignment (ResultSet)
n should be set when the assignment is first created by professor
T5: (2) Display questions added on the GUI
(U6) P: (3) I want students to be able to submit their assignments
Dependent on U4
T1: (2) Student can input answers for each question (GUI)
T2: (1) Students answers can be compared with true answer

(U7) S: (1) I want to be able to see my mark as soon as possible so I don't need to go check it later
T1: (6) Set up Student Assignment table that stores student marks for each individual Assignment
T2: (1) Display total result of assignments right away

T3: (1) store result of assignment
Store only if the current mark exceeds the current stored value

(U8) S: (1) I want to be able to view the class average, to compare my own grades to the class
T1: (4) Determine class average for assignment using the database
T2: (4) Display the class average for an assignment on the GUI

Non-user story tasks (NST):
T1: (6) Create table for students
T2: (5) Validators for Professor entering students
T3: (4) Validators for parsing data files
T4: (4) Add Tables to Questions
T5: (4) Add Tables to Assignments
T6: (5) Validators for Professor entering questions
T7: (5) Validators for Professor entering assignments
T8: (5) Unit Testing DOA functions
T9: (3) Display deadline of an assignment on student GUI

Not yet assigned User stories
P: I want the assignment to be individualized
P: I want grades to be computed and uploaded automatically
P: I want to be able to maintain (delete/edit) an assignment
P: I want students to be able to retry assignments
S: I want to be able to retry my assignments to get higher grades
P: I want to make assignments visible to specific students
P: I want to be able to create complex problems without LaTEX or PERL where I can use mathematical symbols such as summation.