Project Execution

Team: L02_04

Alley Dismay, Ka-Kit Jason Cheung, Pak Sun (Harrison) Fok, Zenan (Will) Ji, Zongyang (Eddie) Li

November 6, 2017

Table of Contents

Product Backlog (Version 1)

Personas

Student Personas:

Sara Bai

- 20 year-old student, female, Canadian-born-Chinese, in 2nd year at UTSC.
- Speaks English fluently.
- Specialist in Sociology with a 2.5 CGPA.
- Loves to read long papers and write essays, but hates math.
- Low technological competence, only uses her computer to watch videos on YouTube, basic usage of MS Word to write essays and uses default web browser on computer to check UTSC Portal.
- Uses a very basic Nokia phone to make phone calls and text.
- She always prints her homework off the computer because she hates reading off a computer screen.
- Takes STAB22 as an elective, since she was told it was an easy A.
- Doesn't want to spend too much time learning software for an elective course, just to complete assignments.
- Wants to just use her uToronto credentials as opposed to making a new account as it would be too much work.
- Likes to divide her assignments and homework into several parts to complete them in smaller chunks, over time.
- Would rather do hardcopy assignments because she believes using software will complicate things.
- Likes to be prepared and caught up for every lecture and before doing every assignment.

Lucas Liu

- 20 year-old student, male, from China, in 3rd year at UTSC.
- Speaks English and Chinese fluently.
- Double major in Statistics and Physical Sciences
- Introvert and autistic, never wants to communicate with people and enjoys staying alone in a room.
- Good computer skills: uses applications like the Microsoft Suite, Dropbox, Skype, computer games
- Typically too lazy to do homework because it's too boring and not engaging enough, would rather play video games like Dota 2
- Very interested in exploring new software and reading about new technologies.
- Works 25 hours a week at a retail store, trying hard to balance work and school.
- Likes to do a lot of practice and do as many problem sets as he can to prepare for tests and understand material
- Hates when professor give limited examples to work on
- Wants to be able to redo assignments to get the best possible mark.
- Appreciates how software looks and loves to use aesthetically pleasing software.
- Doesn't always have internet connection though, so prefers to have a desktop app rather than web app

Alice Smith

- 22 year-old student, female, single Irish-Canadian.
- Lives on residence, with constant access to the school's wifi.
- Speaks English and Irish Gaelic.
- Specialist in Statistics, learns new things quickly and has strong multitasking abilities.
- Passionate about academics, outgoing and spends her free time getting her work done as soon as possible, tries to learn the fastest way of doing everything.
- Loves technology and has access to the cutting-edge of technology.
- Proficient in MS Suite, Photoshop, Final Cut Pro, and numerous other softwares.
- Only likes to do assignments once and disregards them afterwards, not looking at them again.
- Gets frustrated easily when it takes multiple steps to do something simple.
- Is a visual learner, so likes to look at graphs and charts which will help her learn the material better.
- Likes to work on mobile devices like her phone or tablet because its more convenient for her to carry around, rather than a desktop or laptop.
- Likes to keep track of her performance in assignments and other course work she has done.

Professor Personas:

Professor Obi Wan-Too-Thrie

- 65 year-old professor, female
- Speaks English and Hindi
- Works as a professor in the Statistics Department where she mainly teaches introductory statistics courses.
- Prefers to use simple software that is straight forward and uncomplicated.
- She can only perform tasks when someone teaches her and follows the exact same steps
- Little exposure to technology, no cell phone and uses very basic functions in MS Word and Excel to enter grades of her students.
- Teaching style: lets students practice on their own to figure out their own method of approaching problems.
- Doesn't use software that is not engaging or interesting aesthetically
- Currently has students print out exercises to do and hand in at tutorial, but those take really long to be marked (2-3 weeks)
- Teaches classes as large as 1000 students and is usually unable to go over exercises in lecture
- Wants to be able to generate many different questions and types for students through the software, so that students can work them out at home.
- Wants software to be intuitive and familiar to the current software she uses, like MS Word and Excel.
- Believes that gamification(virtual rewards) of exercises/ assignments will increase student's interest in learning and course content.

Professor Bob

- 35 year old stats professor, male with no wife or kids
- works at a the university of Toronto with a degree in both Computer Science and Statistics as a Professor and researcher
- Prof Bob is a full time professor that teaches multiple first and second year courses on Statistics
- Likes to be very organized, likes predictability and things to be done quickly and efficiently
- Speaks English only
- Used to use software for assigning and having students answer questions, but it has become outdated, confusing to use and didn't have sufficient functionality (adding graphs, attachments to questions), which made him frustrated and stopped using it
- good computer skills in the Microsoft Suite, multiple IDEs and terminal
- Has at most 100 students in each class and only a couple of TAs to mark course work since Prof Bob is constantly busy doing research and writing papers
- wants to be able to quickly and easily create and upload questions and distribute them amongst all students
- wants students to be able to provide immediate feedback once students finish an assignment
- Likes to see how the class is doing and what material they are struggling with
- his teaching style is that he prefers students learn by example, rather than memorizing theorems, hoping using the system will allow for abundant question generation to provide many examples to learn and prepare with for exams
- want to give students as much practice / retries on different assignment questions until the deadline of the assignment

User Stories

- As a Professor (Obi-Wan-Too-Thrie), I want to be able to create different types of questions like multiple choice, fill-in-the-blanks, matching, etc.
- As a Professor (Obi-Wan-Too-Thrie), I want each student's version of each question to be different, by having randomized numerical values.
- As a Professor (Bob), I want to assign graphics to questions.
- As a Professor (Bob), I want to be able to add/edit/delete questions from my csv.
- As a Professor (Bob), I want to be able to select questions from the csv and create an assignment (a collection of questions) for the students to answer.
- As a Professor (Bob), I want to view/edit/remove questions from my csv after I've created them.
- As a Professor (Obi-Wan-Too-Thrie), I want to register a professor profile.
- As a student (Sara), I want to be able to make a student profile.
- As a Professor (Bob), I want toggle the visibility of assignments for students, so that students won't see drafts of the assignments.
- As a Professor (Bob), I want to set a timeframe for when assignments can be completed. (i.e. If it's 1 week of visibility. At the end of 1 week, the assignment's visibility is automatically turned off).
- As a student (Alice), I want to be able to see a list of assignments that I have completed and ones that I have not yet completed.
- As a student (Sara), I want to know the deadline of each posted homework.
- As a student (Sara), I want to view which homework is graded.
- As a student (Lucas), I want to be able to answer the questions using this system and submit the assignment.
- As a student (Sara), I want the online homework system to allow me to save my current answer and quit an unfinished homework.
- As a Professor (Obi-Wan-Too-Thrie), I want a set of questions belonging to an assignment to be automatically marked as soon as a student completes it, so that they get immediate feedback.
- As a professor (Bob), I want to be able to store the student's highest, most recent grade on an assignment in the csv.
- As a student (Sara), I want to know the mark of all the tests and the solution of the last test done.
- As a Professor (Bob), I want to see how many students get each question right or wrong, so that I can gauge the class performance.

- As a Professor (Bob), I want assignment feedback to consist a list of correctly and incorrectly answered questions, the number of questions answered correctly in percentage and also the correct answers for questions that were answered incorrectly.
- As a student (Lucas), I want to be able to redo assignments with new questions until the deadline so that I can get the most practice on a topic.
- As a Professor (Obi-Wan-Too-Thrie), I want one occurrence of each selected question in a single assignment.
- As a Professor (Obi-Wan-Too-Thrie), I want an assignment to re-generate random values each time a student attempts an assignment.
- As a Professor (Bob), I want to be able to create multiple lectures for each class I teach, if I need to.
- As a Professor (Obi-Wan-Too-Thrie), I want to be able to add students to a class, so that they can view and complete any assignment questions in the course.
- As a student (Sara), I want to login to the online homework system using my UTORid and password.
- As a Professor (Obi-Wan-Too-Thrie), I want to create badges and rewards for students depending on their performance on assignments, so that the gasification aspect increases interest.
- As a student (Alice), I want to receive a notification/email indicating when a new assignment has been uploaded, so that I can finish it as soon as possible.
- As a student (Sara), I want to know the related chapter of each posted homework.
- As a student (Sara), I want to be able to save the assignment as a PDF, and other formats and print it so that I can work outside the app.
- As a student (Lucas), I would like to return to where i stopped when reopen the app.
- As a student (Alice), I want to track and see my progress on an assignment or set of questions with a progress bar.
- As a student (Sara), I want the online homework system to have a tutorial to teach me how to use this system when I first use it.

The change in the Product Backlog

During the sprint scrum at the end of sprint 1, we re-evaluated the number of story points for our unfinished tasks and decomposed new user stories from the product backlog to implement once we finish the unfinished tasks during sprint 2. As we were doing so, when we encountered the user story:

As a Professor (Bob), I want to be able to store all my questions and the corresponding answers in a csv by topic.

As a team, we decided to remove this user story from the product backlog because it would have made more sense to implement this into user story 7 (shown in the sprint plan for Sprint 2) when inserting a question into the .csv file which holds all the questions.

In addition to this, we also changed all the user stories that required us to store data in some sort of database to storage in a .csv file. This is because it makes development and working with data easier for us as a team.

Sprint 1

Decomposed User Stories for the first sprint

User Story 1: As a Professor (Bob), I want to register a professor profile. (5 story points)

Tasks:

a) Design a form / window for registering profiles that takes in information for the profile. Possible options for fields are: **(1 Story point)**
   i) Email
   ii) First name / Last Name / Full Name
   iii) Have a submit button for the user to process the action

b) Implement Professor class to generate professor objects that store the information collected by the form and has methods to do things like. (helper functions to write and retrieve professor information to a csv.) **(2 Story points)**

c) Set up the form to, when the submit button is clicked, to take the information in the form and generate a corresponding Professor object. **(1 story point)**
   Dependency: a, b
   i) implement the back-end methods to create a professor object with the filled in values of the form, and insert it to the csv.

d) Create a form / window to read the information of a professor object and display it. And likely allow editing. Perhaps repurpose the Profile Registration form. **(1 Story point)**
   Dependence: a, b, c

User Story 2: As a student (Sara), I want to be able to make a student profile. (4 story points)

Tasks:

a) Split the general profile form into two versions **(1 story point).**
   i) create a user interface similar to the professor one with basic fields of name, email and student number for students
   ii) when a student is registering, show only the student specific fields required to make a student profile.
   iii) use the Tkinter framework to develop the UI with the required fields and buttons.

b) Create a option menu with Professor and Student profile choices so the user can select what kind of profile they're making. **(1 story point)**

c) Create a super class that student and professor objects inherit from. **(1 story point)**
   i) have an attribute that indicates whether or not the user is a professor or student.
   ii) implement student and parent classes and allow them to inherit from the super class called User.

d) Take the values provided in the fields and create a student object, and then insert it into the csv. **(1 story point)**
   Dependency: a, c
   i) create csv helper methods to insert the created student object into the csv given its values.
   ii) Connect the submit button from the form to the back end methods to store the created user into the csv.
   iiii) would be calling the generic user class' insert into csv method.

User Story 3: As a Professor (Bob), I want to be able to create different types of questions like multiple choice, fill-in-the-blanks, matching, etc. (8 story points)

Tasks:

a) Design and implement the UI for letting the professor choose their type of question**. (1 story point)**
   i) Draw wireframe/sketch the UI out on paper
   ii) implement the UI with Tkinter framework
   iii) Have a radio button for each type of question the professor could create.

b) Create a generic question class that will hold the general attributes of a question. **(1 story point).**
   i) each type of question will be a subclass that inherits from this class.
   ii) this general question class has attributes like a boolean value for if the question was answered correctly, a field of the question and answer for the particular question.
   iii) generic question class should have a method that inserts the created question into the csv.

c) Add the corresponding subclasses for multiple choice, fill-in-the-blanks and matching questions. **(1 story point)**
   i) implement these classes with their question specific fields and have it inherit from the generic question class.
   ii) for mc have an array to hold the options, match questions have a dictionary to hold the key:value, question:answer pairs, fill-in-the-blanks can use a box # : answer key:value pair as well.

d) Design and implement the forms/UI for each different type of question (mc, fill-in-the-blanks and matching questions) for the professor to create a question with. **(3 story points)**
   i) draw wireframe/sketch the UI out on paper
   ii) implement the UI with Tkinter framework
   iii) Do this for each type of question(ie. Multiple choice, fill-in-the-blanks, matching)

e) connect the front end to the back end methods. **(2 story points).**
   Dependency: a, b, c

User Story 4: As a Professor (Bob), I want each student's version of each question to be different, by having randomized numerical values (5 story points)

Tasks:

a) Implement a random function to randomize the values of a question each time it's being called. **(3 story points)**

i) implement a method to take the upper and lower bounds of a random number to be generated.

ii) make a call to this randomize value function each time a question is retrieved from the csv that stores the questions.

b) Return a question object that has the newly generated random values to be used by the professor. **(1 story point)**

i) Create a new question object that has these random values as a part of the object's attributes.

c) Implement UI to allow the professor to choose a custom range of randomize values for a particular question and connect the back-end to front-end. **(1 story point)**

Dependency: a, b

i) wireframe/sketch the UI with text fields to input a upper and lower bound.

ii) the menu should display the question that will have the randomized values. (use the csv select helper methods)

iii) have a submit button for the professor to submit his option

iv) connect the front and back-end methods

User Story 5: As a Professor (Bob), I want to be able to select questions from the csv and create an assignment (a collection of questions) for the students to answer. (8 story points)

Tasks:

a) create an assignment class that has attributes and methods for an assignment. **(1 story point)**

i) have a field indicating the number of questions selected for the assignment.

ii) have a list to hold the chosen questions in an assignment.

iii) attribute to indicate deadline and start date

b) Implement method to create, remove questions, which will be the assignment. **(2 story point)**

Dependency: a

i) implement a method for the assignment class use the csv select helper to retrieve questions from the csv and append it to a list that will hold the questions for an assignment.

c) create UI and implement method that displays all the questions in the csv that the professor can choose from to add to an assignment. **(1 story point)**

Dependency: a, b

i) wireframe a UI to display all questions that are present in the csv.

ii) create a method to select all from csv and print to the screen.

iii) link button to the method implemented in b)

d) Create UI to display an individual assignment with all of its questions. **(1 story point)**

Dependency: a, b, c

i) implement method to print out all values in the array holding the question to an assignment.

e) Implement csv methods for assignment class. **(3 story points)**

Dependency: a

i) create model/table in the csv to hold the the attributes created in the assignment class.

ii) create insertion, retrieval methods for an assignment object into the csv.

Sprint Plan

Team of 5 developers: Andy, Eddie, Harrison, Jason, Will:

- Andy is able to complete 7 story points a week
- Eddie - 6 story points a week
- Harrison - 6 story points a week
- Jason - 5 story points a week
- Will - 6 story points a week

Developers will work for 7 days a week and the length of the sprint is 1 week

Team Velocity

The team velocity will be 30 story points a week.

Provisional Schedule

Tasks	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7
1a	Andy:1						
1b		Andy:2					
1c			Andy:1				
1d				Andy:1			
2a			Jason:1				
2b				Jason:1			
2c					Jason:1		
2d						Jason:1	

3a	Eddie: 1						
3b		Eddie:1					
3c			Eddie:1				
3d				Eddie:1			
3e					Eddie :2		
4a		Will：3					
4b			will：1				
4c					will：1		
5a	Harrison: 1						
5b	Harrison: 2						
5c		Harrison: 1					
5d			Harrison: 1				
5e			Harrison: 1	Harrison: 2			

Based on this provisional schedule, 4 user stories will be completed by the end of the sprint, with 28 user stories burned.

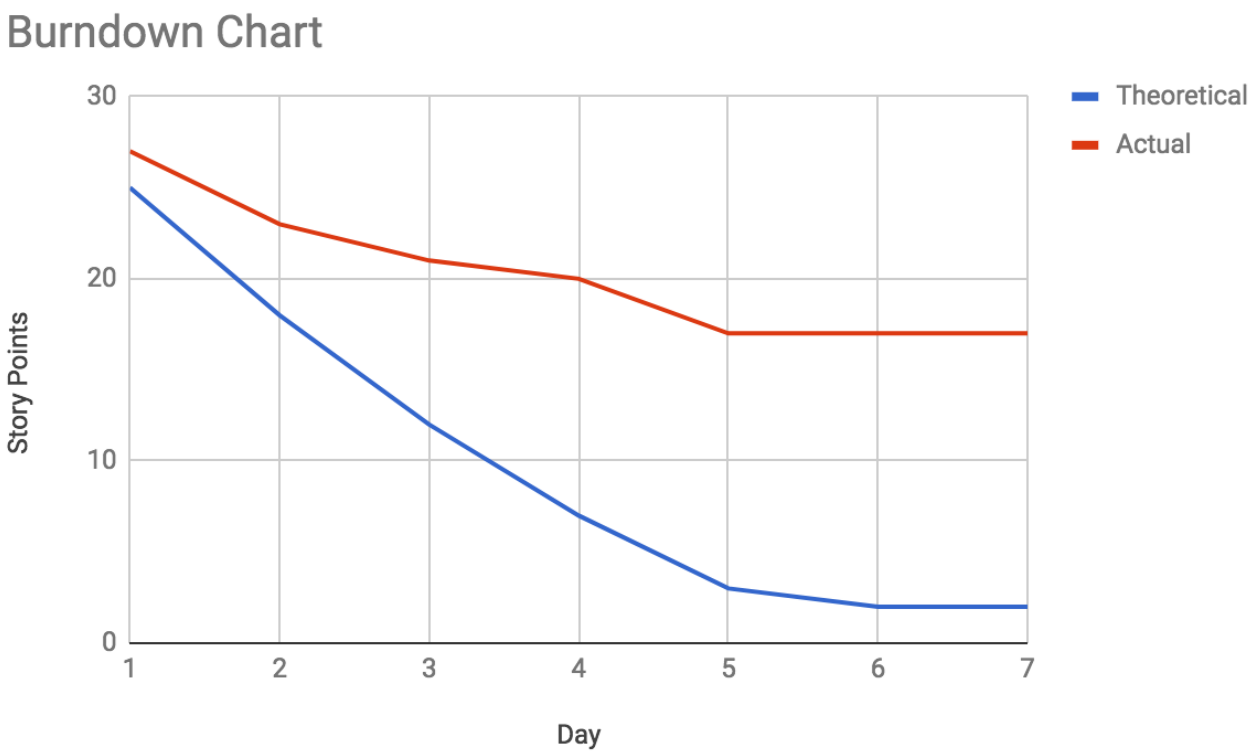
Sprint Report

Tasks	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7
1a	Andy:1						
1b	Andy:2						
1c		Andy:5					
1d			Andy:5				
2a							
2b							
2c							
2d							
3a	Eddie: 1						
3b		Eddie:1					
3c			Eddie:2				
3d				Eddie:2			
3e					Eddie :2		
4a		Will:3					
4b							
4c							

5a	Harrison:1						
5b	Harrison:2						
5c	Harrison:1						
5d	Harrison:1						
5e		Harrison:2	Harrison:3				

At the end of this sprint only user story 3 was completed, while the other tasks required more time than expected to complete. As a result, these tasks will be completed in sprint 2.

Burndown Chart

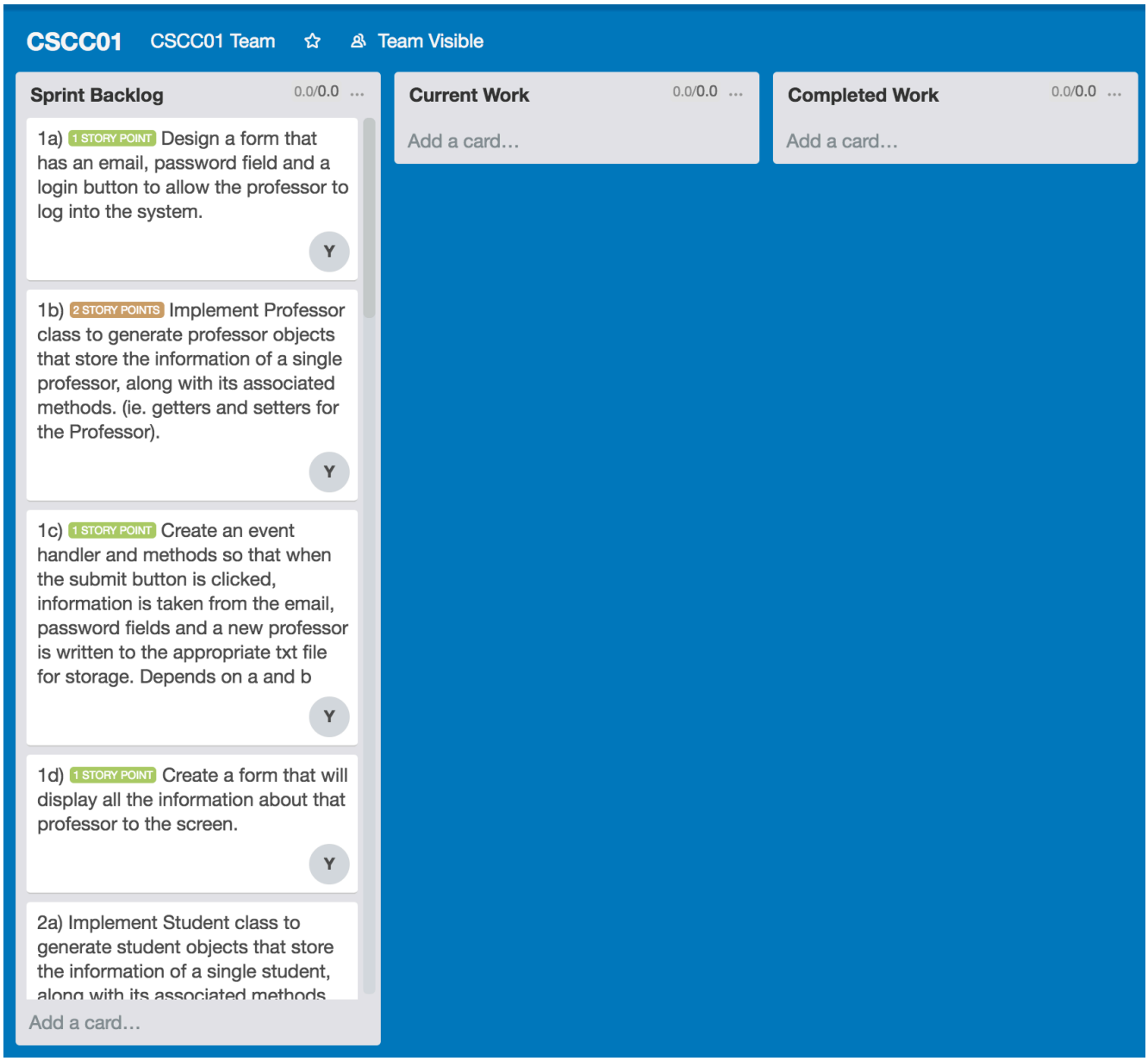


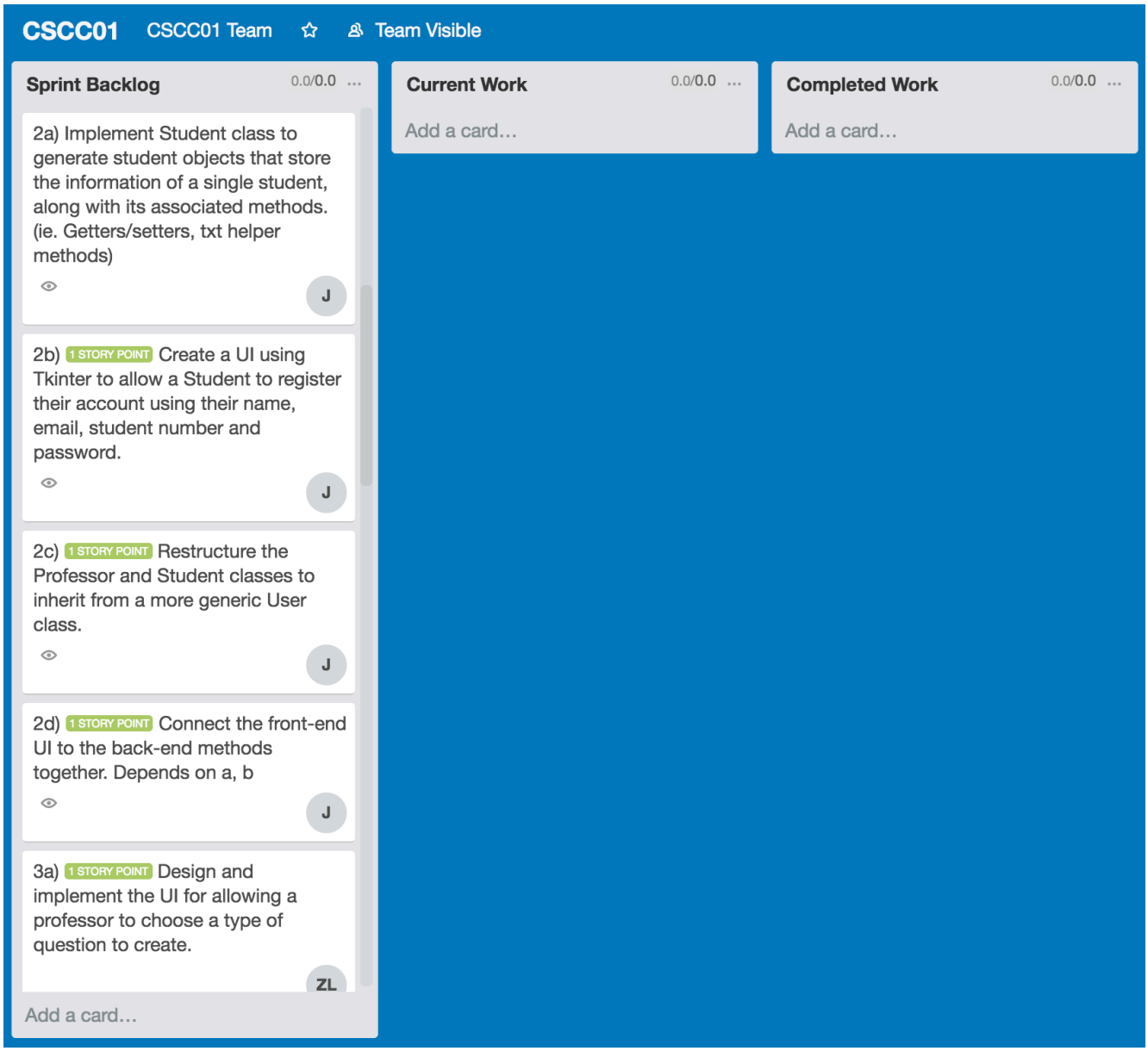
Snapshots of Taskboard

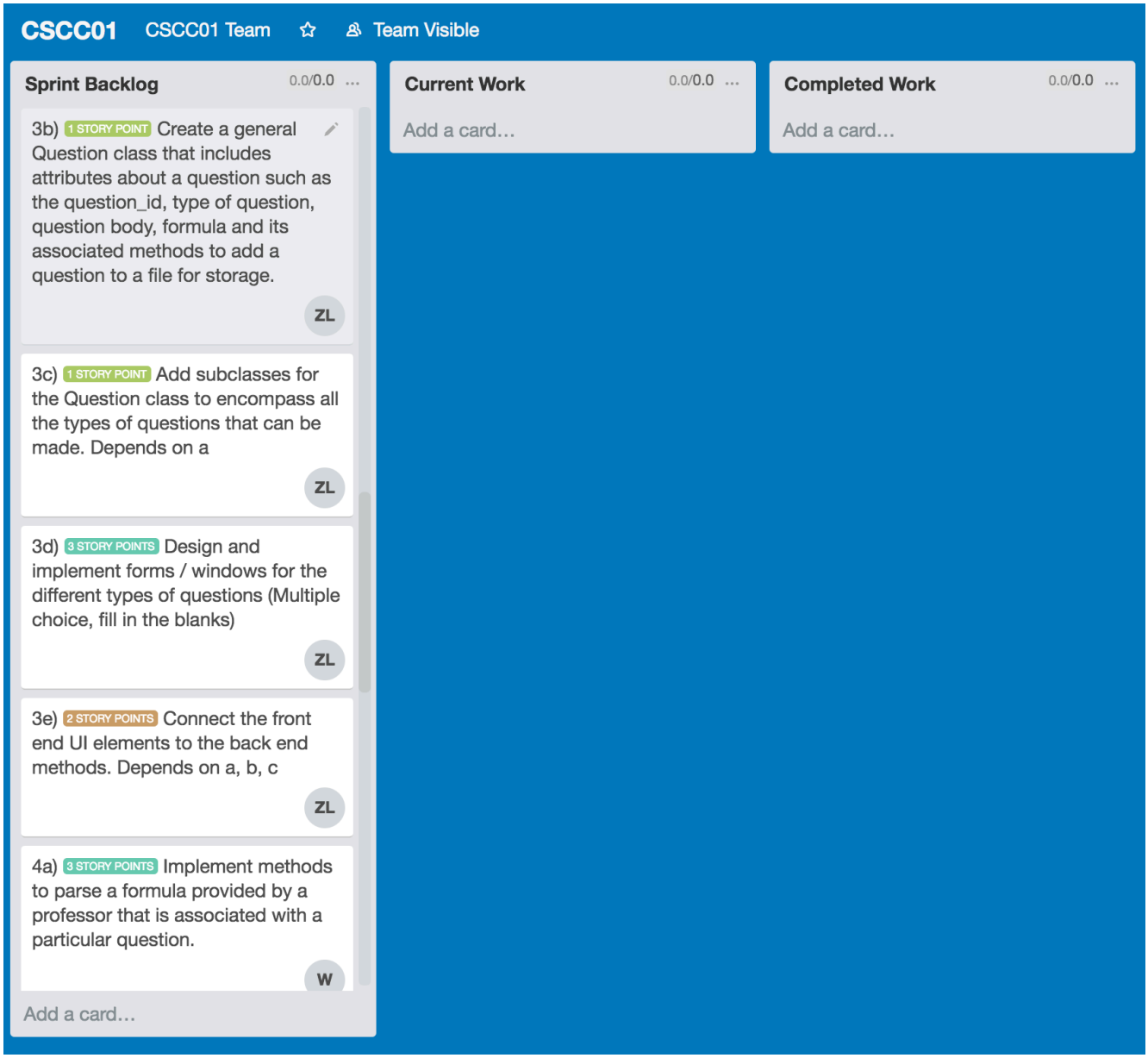
Since the Trello board was filled with too many tasks, 2 screenshots were not sufficient to capture the entire taskboard.

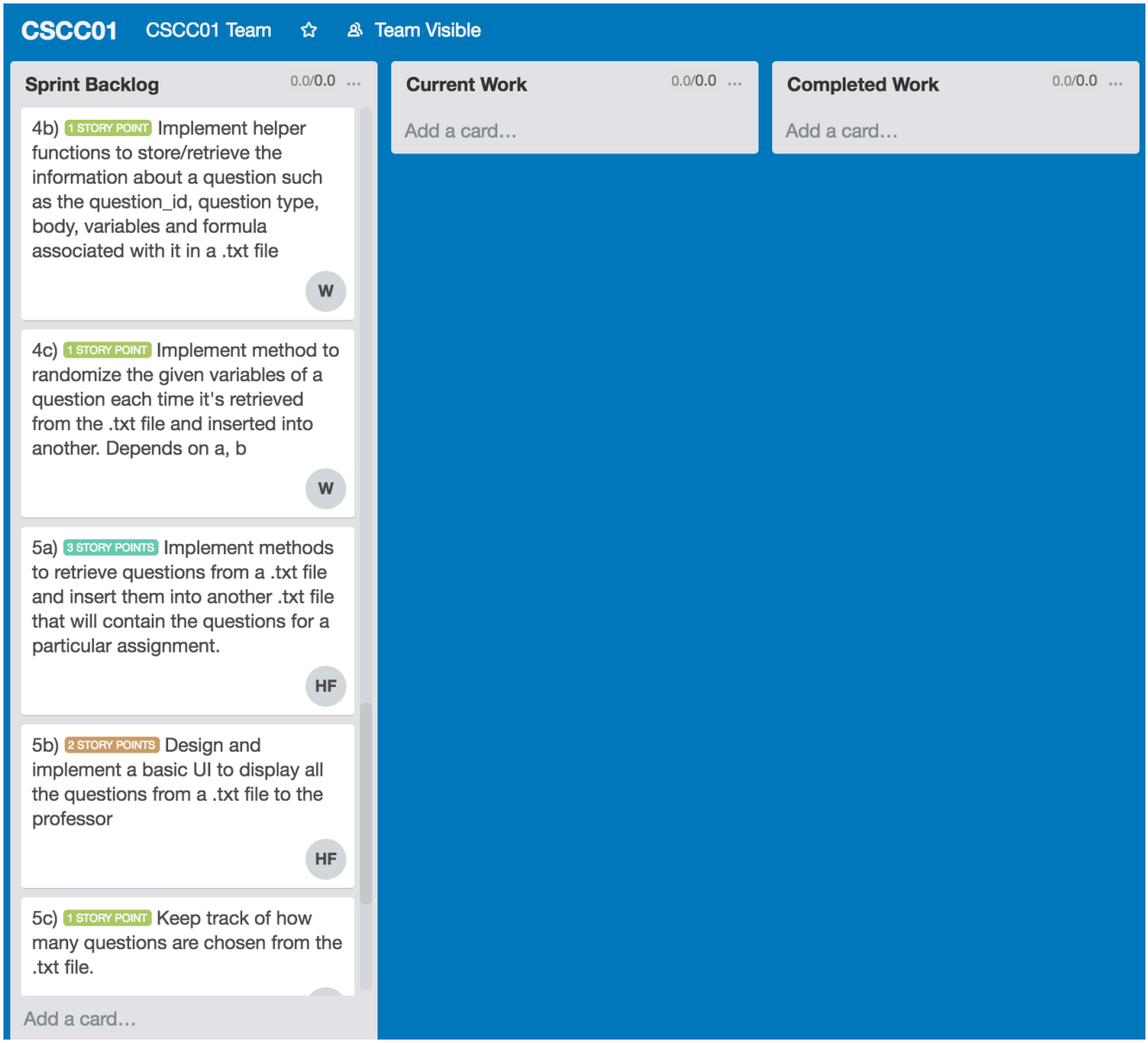
Start of Sprint

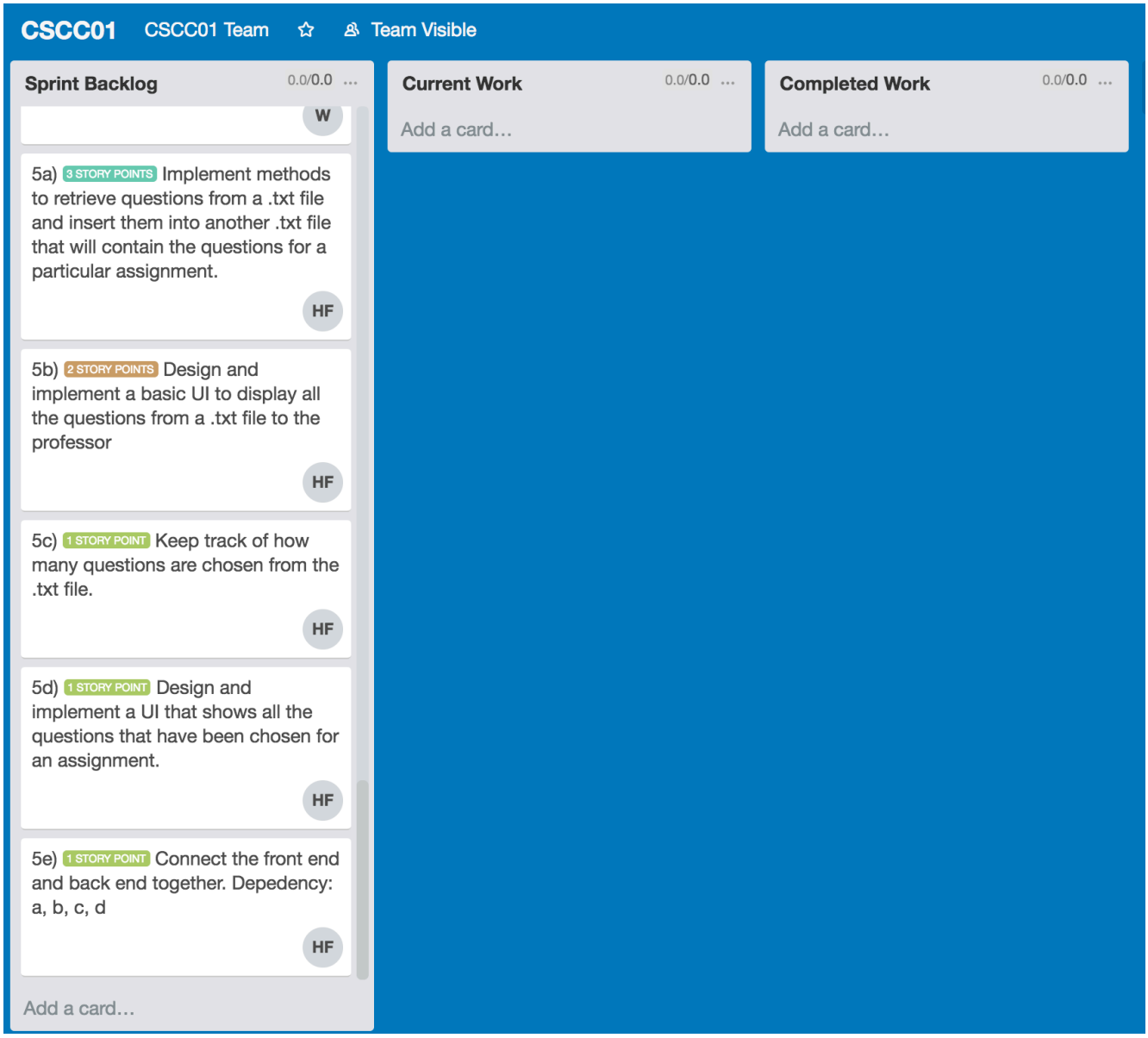
Each list has the task number, number of story points and the person responsible for the task. For example, the first one is task 1a, which requires 1 story point and is allocated to Andy.
Y = Andy, J = Jason, ZL = Eddie, W = William, HF = Harrison



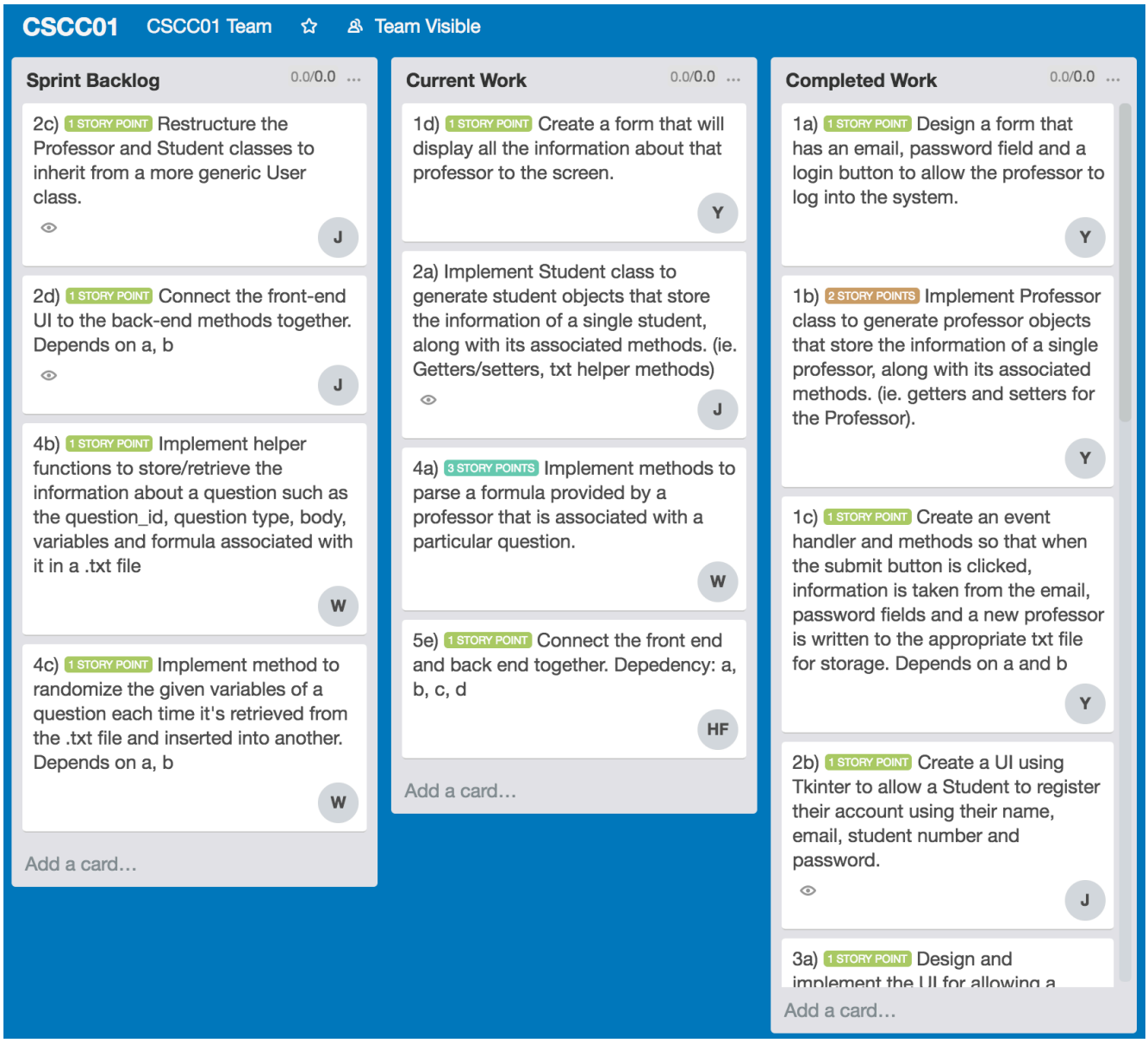

CSCC01
CSCC01 Team
Team Visible
Sprint Backlog
0.0/0.0
2a) Implement Student class to generate student objects that store the information of a single student, along with its associated methods. (ie. Getters/setters, txt helper methods)
J
2b) 1 STORY POINT Create a UI using Tkinter to allow a Student to register their account using their name, email, student number and password.
J
2c) 1 STORY POINT Restructure the Professor and Student classes to inherit from a more generic User class.
J
2d) 1 STORY POINT Connect the front-end UI to the back-end methods together. Depends on a, b
J
3a) 1 STORY POINT Design and implement the UI for allowing a professor to choose a type of question to create.
ZL
Add a card…
Current Work
0.0/0.0
Add a card…
Completed Work
0.0/0.0
Add a card…

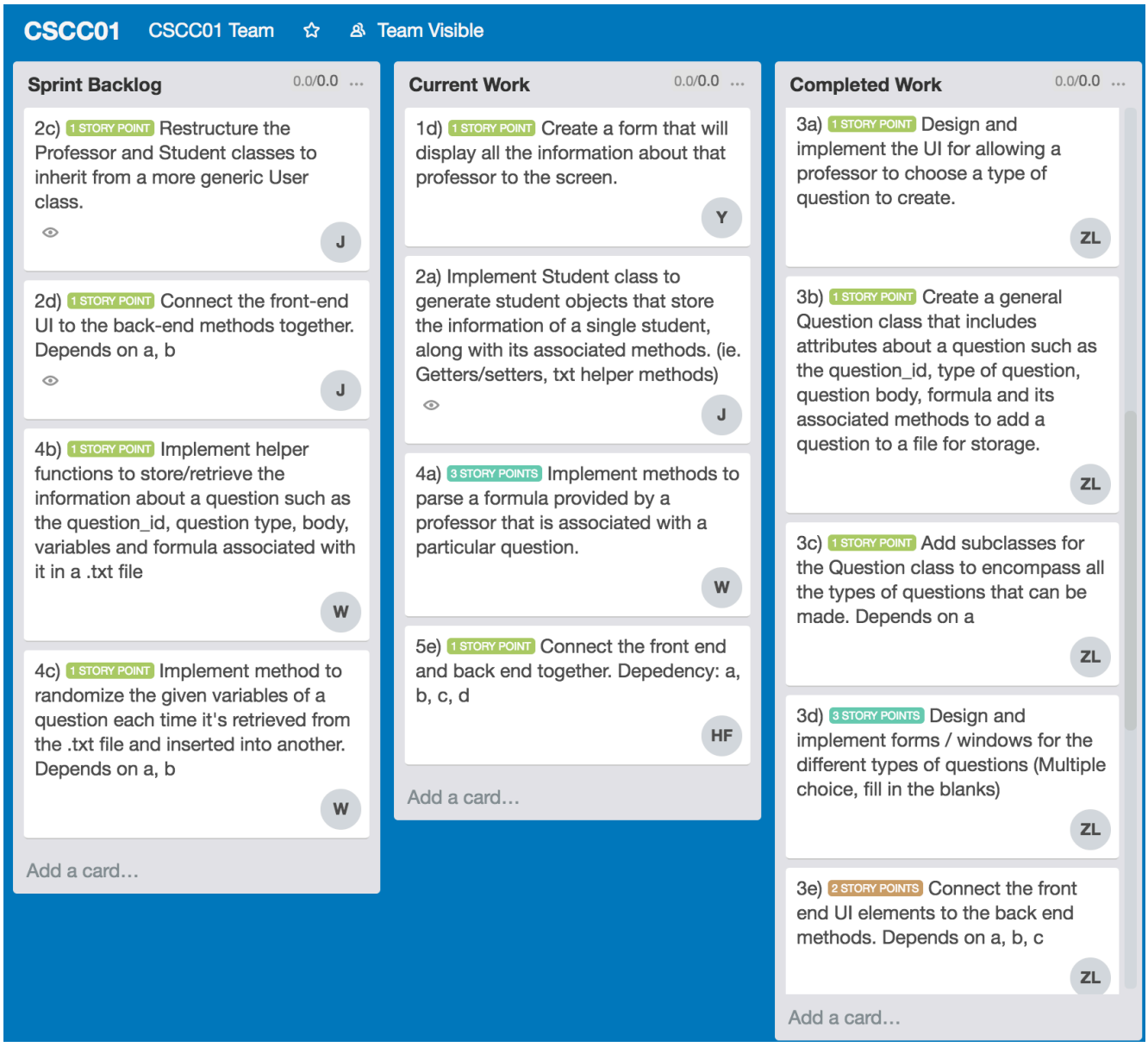

CSCC01
CSCC01 Team
Team Visible
Sprint Backlog
0.0/0.0
3b) 1 STORY POINT Create a general Question class that includes attributes about a question such as the question_id, type of question, question body, formula and its associated methods to add a question to a file for storage.
ZL
3c) 1 STORY POINT Add subclasses for the Question class to encompass all the types of questions that can be made. Depends on a
ZL
3d) 3 STORY POINTS Design and implement forms / windows for the different types of questions (Multiple choice, fill in the blanks)
ZL
3e) 2 STORY POINTS Connect the front end UI elements to the back end methods. Depends on a, b, c
ZL
4a) 3 STORY POINTS Implement methods to parse a formula provided by a professor that is associated with a particular question.
W
Add a card…
Current Work
0.0/0.0
Add a card…
Completed Work
0.0/0.0
Add a card…

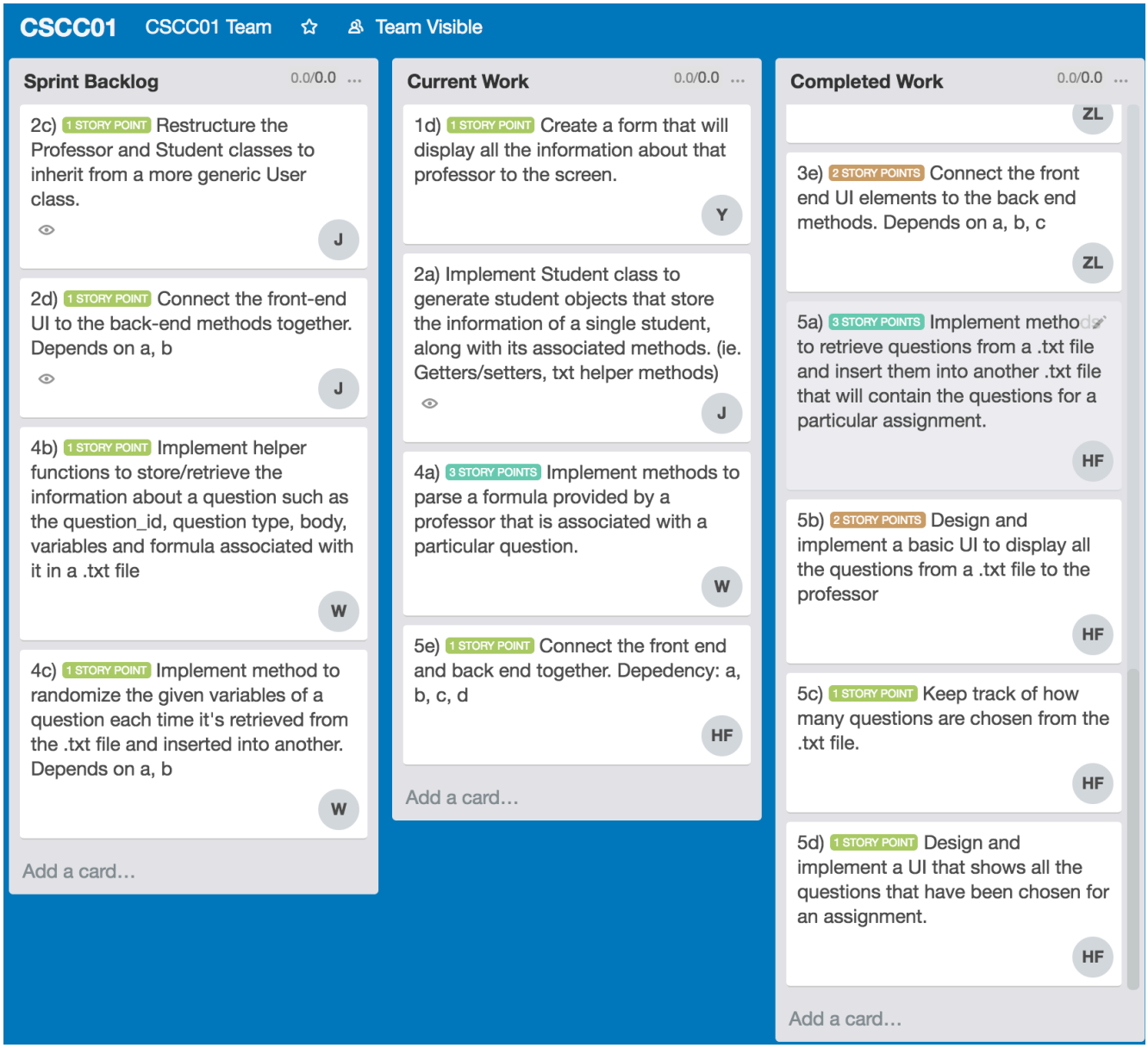

CSCC01
CSCC01 Team
Team Visible
Sprint Backlog
0.0/0.0
4b) 1 STORY POINT Implement helper functions to store/retrieve the information about a question such as the question_id, question type, body, variables and formula associated with it in a .txt file
W
4c) 1 STORY POINT Implement method to randomize the given variables of a question each time it's retrieved from the .txt file and inserted into another. Depends on a, b
W
5a) 3 STORY POINTS Implement methods to retrieve questions from a .txt file and insert them into another .txt file that will contain the questions for a particular assignment.
HF
5b) 2 STORY POINTS Design and implement a basic UI to display all the questions from a .txt file to the professor
HF
5c) 1 STORY POINT Keep track of how many questions are chosen from the .txt file.
Add a card…
Current Work
0.0/0.0
Add a card…
Completed Work
0.0/0.0
Add a card…


CSCC01
CSCC01 Team
Team Visible
Sprint Backlog
0.0/0.0
Current Work
0.0/0.0
Add a card…
Completed Work
0.0/0.0
Add a card…
W
5a) 3 STORY POINTS Implement methods to retrieve questions from a .txt file and insert them into another .txt file that will contain the questions for a particular assignment.
HF
5b) 2 STORY POINTS Design and implement a basic UI to display all the questions from a .txt file to the professor
HF
5c) 1 STORY POINT Keep track of how many questions are chosen from the .txt file.
HF
5d) 1 STORY POINT Design and implement a UI that shows all the questions that have been chosen for an assignment.
HF
5e) 1 STORY POINT Connect the front end and back end together. Depedency: a, b, c, d
HF
Add a card…

End of Sprint




CSCC01
CSCC01 Team
Team Visible
Sprint Backlog
0.0/0.0
2c) 1 STORY POINT Restructure the Professor and Student classes to inherit from a more generic User class.
J
2d) 1 STORY POINT Connect the front-end UI to the back-end methods together. Depends on a, b
J
4b) 1 STORY POINT Implement helper functions to store/retrieve the information about a question such as the question_id, question type, body, variables and formula associated with it in a .txt file
W
4c) 1 STORY POINT Implement method to randomize the given variables of a question each time it's retrieved from the .txt file and inserted into another. Depends on a, b
W
Add a card…
Current Work
0.0/0.0
1d) 1 STORY POINT Create a form that will display all the information about that professor to the screen.
Y
2a) Implement Student class to generate student objects that store the information of a single student, along with its associated methods. (ie. Getters/setters, txt helper methods)
J
4a) 3 STORY POINTS Implement methods to parse a formula provided by a professor that is associated with a particular question.
W
5e) 1 STORY POINT Connect the front end and back end together. Depedency: a, b, c, d
HF
Add a card…
Completed Work
0.0/0.0
3a) 1 STORY POINT Design and implement the UI for allowing a professor to choose a type of question to create.
ZL
3b) 1 STORY POINT Create a general Question class that includes attributes about a question such as the question_id, type of question, question body, formula and its associated methods to add a question to a file for storage.
ZL
3c) 1 STORY POINT Add subclasses for the Question class to encompass all the types of questions that can be made. Depends on a
ZL
3d) 3 STORY POINTS Design and implement forms / windows for the different types of questions (Multiple choice, fill in the blanks)
ZL
3e) 2 STORY POINTS Connect the front end UI elements to the back end methods. Depends on a, b, c
ZL
Add a card…


CSCC01
CSCC01 Team
Team Visible
Sprint Backlog
0.0/0.0
2c) 1 STORY POINT Restructure the Professor and Student classes to inherit from a more generic User class.
J
2d) 1 STORY POINT Connect the front-end UI to the back-end methods together. Depends on a, b
J
4b) 1 STORY POINT Implement helper functions to store/retrieve the information about a question such as the question_id, question type, body, variables and formula associated with it in a .txt file
W
4c) 1 STORY POINT Implement method to randomize the given variables of a question each time it's retrieved from the .txt file and inserted into another. Depends on a, b
W
Add a card…
Current Work
0.0/0.0
1d) 1 STORY POINT Create a form that will display all the information about that professor to the screen.
Y
2a) Implement Student class to generate student objects that store the information of a single student, along with its associated methods. (ie. Getters/setters, txt helper methods)
J
4a) 3 STORY POINTS Implement methods to parse a formula provided by a professor that is associated with a particular question.
W
5e) 1 STORY POINT Connect the front end and back end together. Depedency: a, b, c, d
HF
Add a card…
Completed Work
0.0/0.0
ZL
3e) 2 STORY POINTS Connect the front end UI elements to the back end methods. Depends on a, b, c
ZL
5a) 3 STORY POINTS Implement method to retrieve questions from a .txt file and insert them into another .txt file that will contain the questions for a particular assignment.
HF
5b) 2 STORY POINTS Design and implement a basic UI to display all the questions from a .txt file to the professor
HF
5c) 1 STORY POINT Keep track of how many questions are chosen from the .txt file.
HF
5d) 1 STORY POINT Design and implement a UI that shows all the questions that have been chosen for an assignment.
HF
Add a card…

Sprint 2

Decomposed user stories for Sprint 2

The following user stories are an addition to the ones that were not completed during sprint 1.

User Story 6: As a Professor (Bob), I want to add/view/edit/remove questions from my text file after I've created them. (6 story points)

Tasks:

a) Repurpose the Question Creation UI to allow the professor to view, edit and remove questions. (2 story points)
   i. Likely going to be selecting questions from a file that will give a list of questions, so make a window with the questions stacked vertically on top of each other.
   ii. Then use built-in list methods to extract each question in the list and output it in a formatted way.

b) Implement methods to add, edit and remove questions (2 story points)

c) Connect the repurposed front-end UI to the newly implemented back-end method. (2 story points)
Depends on a, b

User Story 7: As a Professor (Bob), I want to be the only one able to add/edit/delete questions from a csv file. (2 story points)

Tasks:

a) Implement a new class OptionsMenu.py that will check whether a user is a professor or student. (2 story points)

i) this class will display different buttons according to what type of user this is. (either student or professor)

ii) if the user is a professor, then provide a button/option to let the professor add/edit/delete questions from the csv.

User Story 8: As a Student (Lucas), I want to be able to answer questions for an assignment using the system and submit my assignment. (6 story points)

Tasks:

a) Design and implement a UI using Tkinter to allow the student to see the questions as a list and have a corresponding text field to type their answer. (2 story points)
   i) also have a submit button that saves all the answers for a student in a .csv file
   ii) after submitting, show the correct answer beside the question if the student got it wrong.

b) implement the back end methods to pull data from the csv file that stores the questions of an assignment and compares the student's answer with the one in the csv when they hit the submit button. (2 story points)
   Dependent on a
   i) store the result of the student's answers from this assignment in a new .csv file called AssignmentAttempts.csv
   ii) connect the back end with the front end.

c) AssignmentAttempts.csv will hold all the results of an assignment, overwriting the respective row with the most recent attempt. (2 story points)
   ie) the .csv file will have the 1st row correspond to the 1st assignment, 2nd row for 2nd assignment and so on…
   Dependent on a, b
   i) if the student retries the 2nd assignment, the 2nd row will be overwritten.

User Story 9: As a Professor (Bob), I want to toggle the visibility of assignments for students, so that students won't see drafts of the assignments. (7 story points)

Tasks:

a) create a button in the AssignmentDashboard GUI to be displayed beside an assignment name that will take you to the 'visibility options menu', with radio buttons for visibility on and visibility off. (2 story points)
   i) create a class for an Assignment that has a boolean attribute for visibility, along with other necessary attributes
   ii) Create a GUI for the student that will display all assignments that have its visibility attribute set to True.
   iii) using tkinter to implement the UI

b) implement the back end methods to pull an assignment from the .csv that stores all the assignments and change it's attribute for visibility depending on what the professor chose. (2 story points)
   i) want to pull the assignment out of the .csv, update it's value for visibility and then insert it back into the .csv that stores all the assignments
   ii) based on this value, the assignments list UI should display the appropriate assignments

c) connect the front and back end together (3 story points)

<u>User Story 10:</u> As a Professor (Bob), I want to set a timeframe for when assignments can be completed. (ie. if the timer is 1 week long, at the end of the week the assignment's visibility will automatically be turned off.) (7 story points)

a) Add a button that takes the AssignmentDashboard GUI menu for the professor beside each assignment. (2 story points)

i) this button will take them to a new SetTimer GUI where they can set the visibility of that assignment for x number or hours/up to a certain date

b) Implement back-end methods so that when the time length is set in the SetTimer GUI, and the timer is up, the visibility will be automatically turned off for an assignment. (3 story points)

i) the next time the program is run and the student opens their dashboard to see the list of assignments, if the time is up then a method will pull the assignment from Assignments.csv, change the attribute for visibility to be false, and the student dashboard will reflect this.

c) Connect the front end to the back end (2 story points).

depends on a,b

User Story 11: As a Professor (Bob), I want to check if the student answers are correct for the assignment once they submit it. (7 story points)

a) Implement methods to iterate through the .csv file to find the particular assignment the student is working on. (2 story points)

i) iterate through each question from the assignment and using the provided formula for the answer, compute the answer.

ii) form a dictionary with the key being the question id and the value being the calculate answer based on the formula.

b) Implement methods to extract the student's answers from AssignmentAttempts.csv, and compare these answers with the calculated answer in part a) (3 story points)

Dependent on a

i) form a dictionary with the key being the question id and the value being the students answer

ii) iterate through both dictionaries from part a and b and compare the values given the same key.

c) Design and implement a UI using Tkinter that will display the results of a students answers when compared to the actual answer. (1 story point)

d) Connect the front-end and the back-end (1 story point)

Dependent on a, b, c

Sprint Plan

Team of 5 developers: Andy, Eddie, Harrison, Jason, Will:

- Andy is able to complete 10 story points a week
- Eddie - 6 story points a week
- Harrison - 8 story points a week
- Jason - 6 story points a week
- Will - 6 story points a week

Developers will work for 7 days a week and the length of the sprint is 1 week

Team Velocity

The team velocity will be 36 story points a week.

Provisional Schedule

This schedule focuses on completing the unfinished tasks from Sprint 1 first, before starting on the sprint 2 tasks.
During the sprint scrum at the end of sprint 1, we decided to switch to .csv files for storage, so user story 3 had to be brought back into the backlog and edited.
For the unfinished user stories, the first one is re-evaluated to 1 task

Day 1: October 30th, Day 7: November 5th

Tasks	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7
1d	Andy:1						
2a	Jason:1						
2c			Jason:1				
2d				Jason:1			

3e	Eddie:2						
4a	Will：3						
4b		will：1					
4c			will：1				
5a	Harrison : 1						
5c		Harrison: 1					
5d			Harrison: 1				
5e			Harrison: 1	Harrison: 2			
6a		Eddie : 2					
6b			Eddie : 2				
6c							
7a	Andy: 1	Andy: 1					
8a					Jason : 2		
8b						Jason : 1	
8c							
9a		Andy: 2					
9b			Andy : 3				

9c				Andy : 2			
10a			Will : 1				
10b							
10c							
11a				Harrison: 2			
11b							
11c							
11d							

By the end of this sprint, we will have finished the unfinished tasks from sprint 1, and will have started working on the tasks for the second sprint.

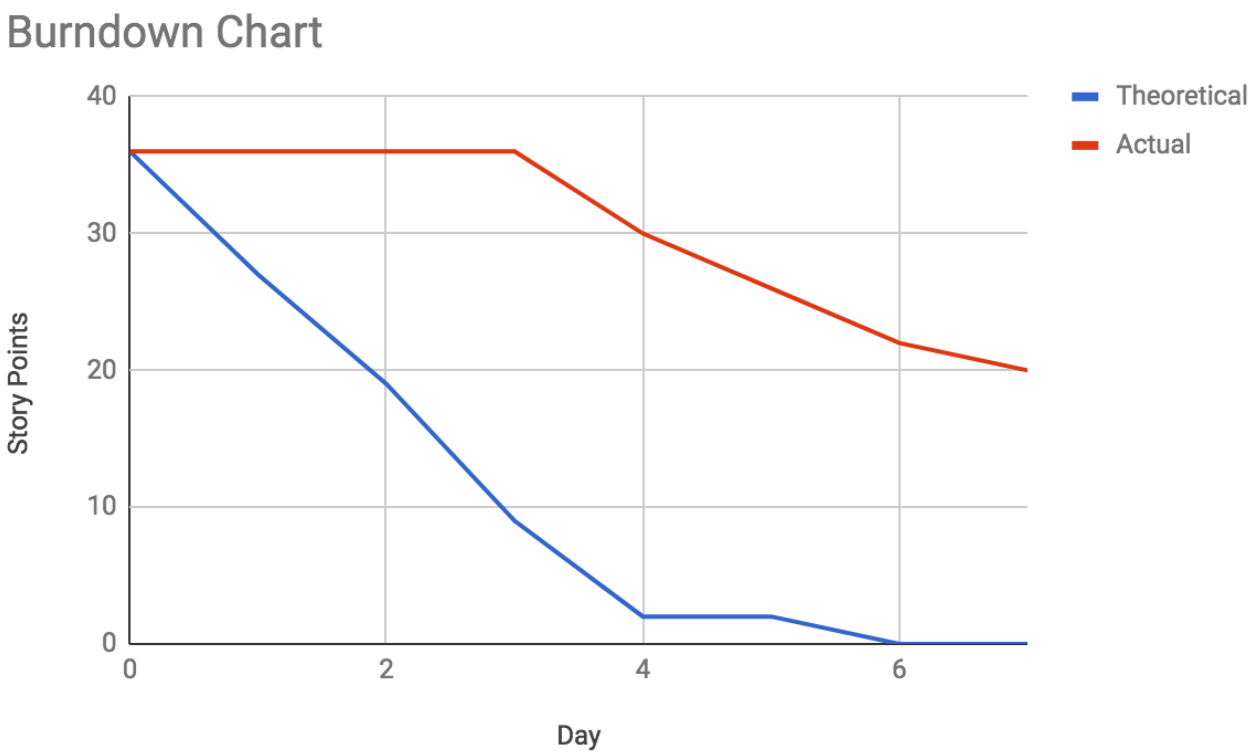
Sprint Report

Tasks	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7
1d			Andy:1	Andy:1	Andy:1	Andy:1	
2a				Jason: 1			
2c				Jason: 1			
2d				Jason: 1			
3e						Eddie:2	
4a			William: 1	William :3			
4b					William: 1		
4c						William:1	
5a							
5c							
5d							

5e					Harrison:2		
6a							
6b							
6c							
7a						Andy:2	
8a							
8b							
8c							
9a							
9b							
9c							
10a							
10b							
10c							
11a							
11b							
11c							
11d							

At the end of this sprint, we finished all unfinished tasks from sprint 1, and started working on the tasks for the second sprint.

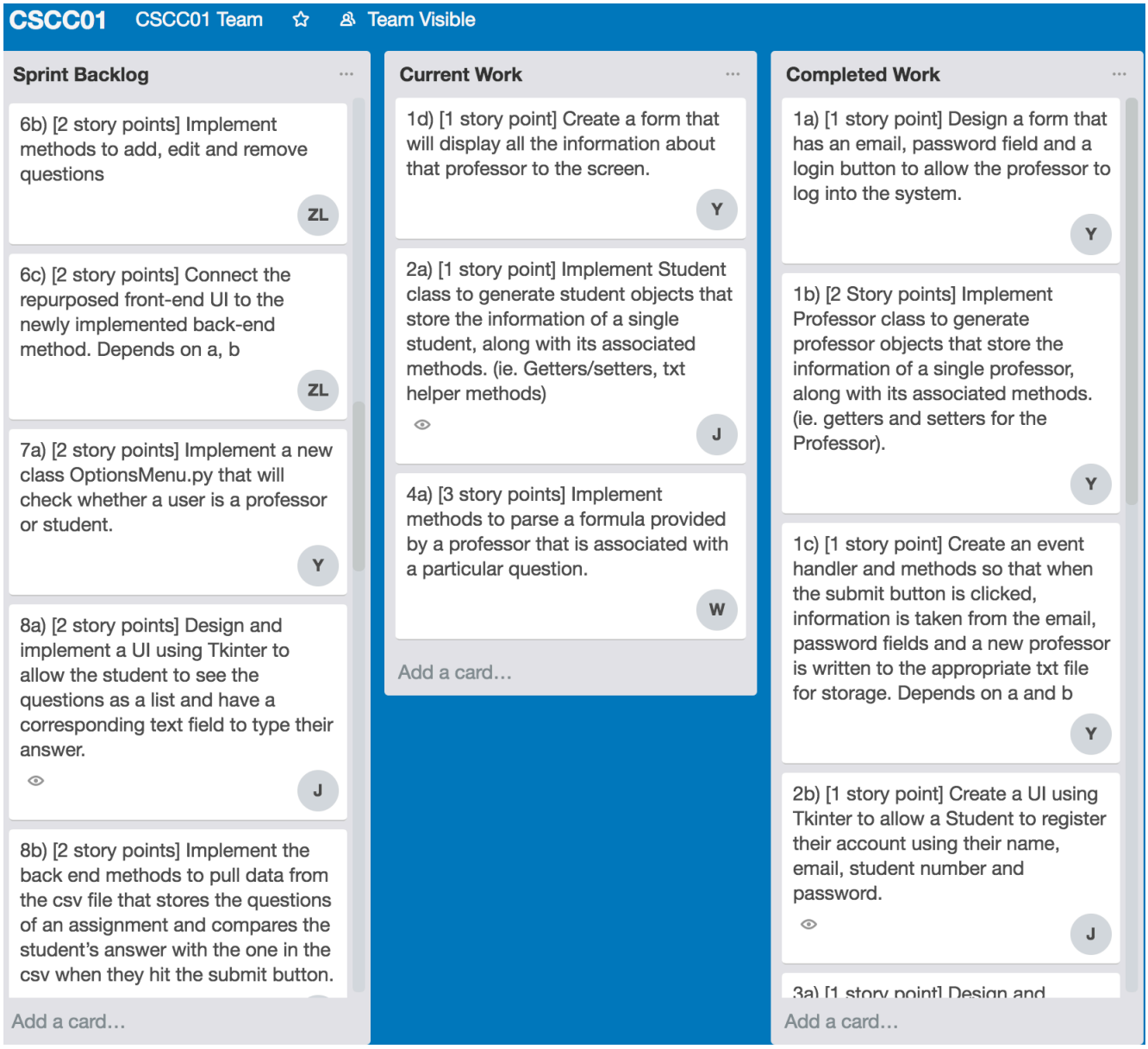
Burndown Chart

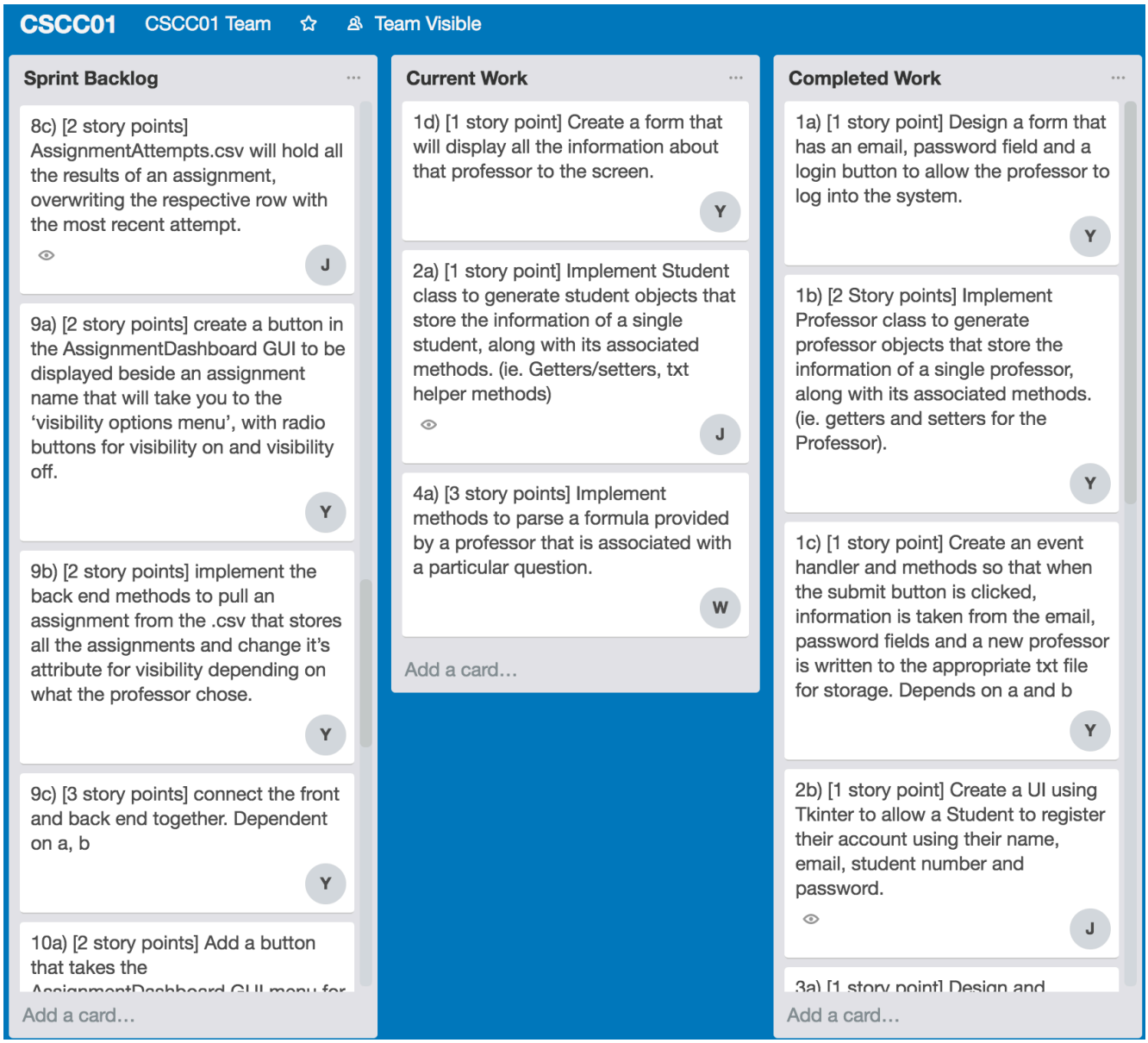


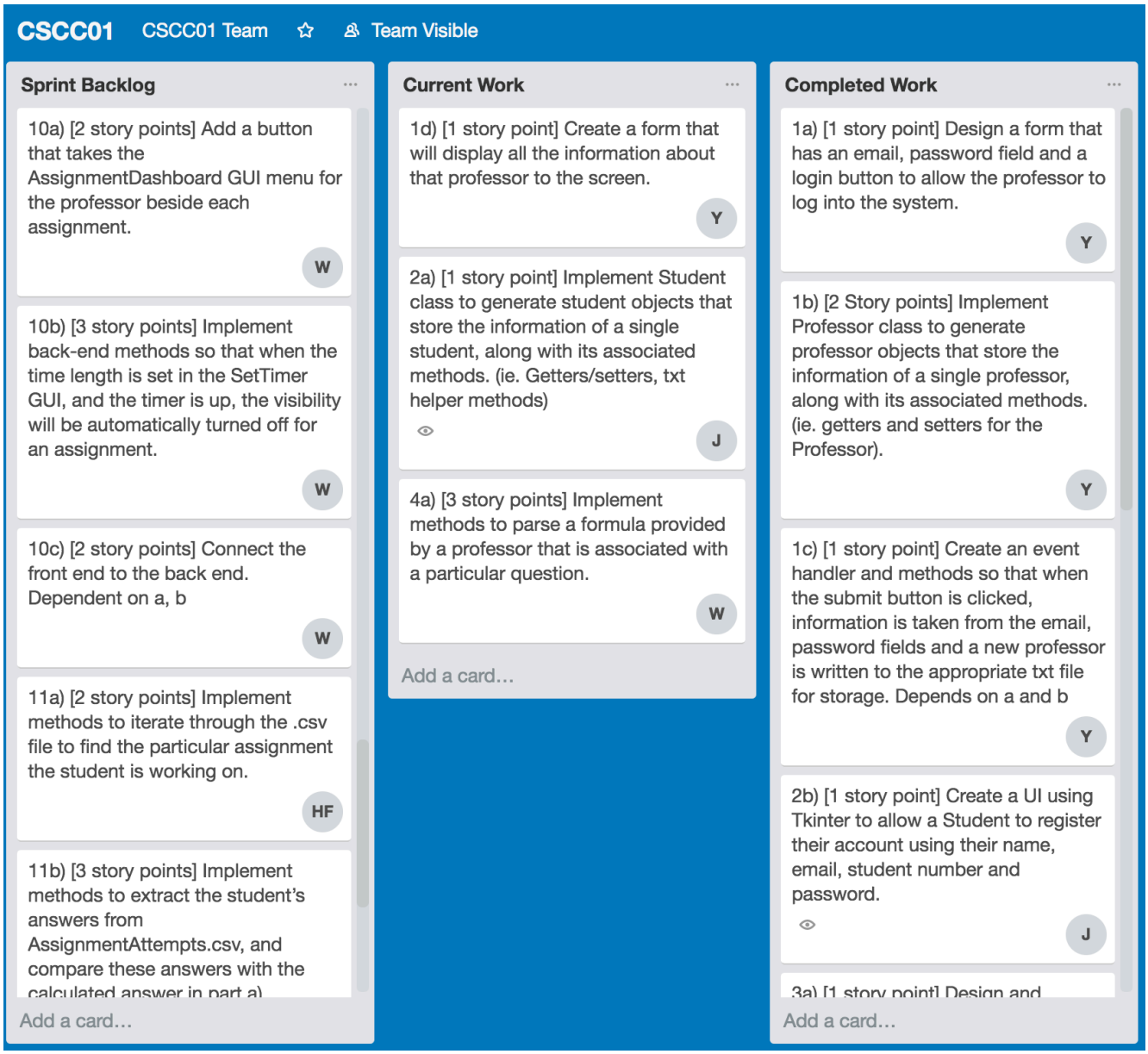
Snapshots of Taskboard

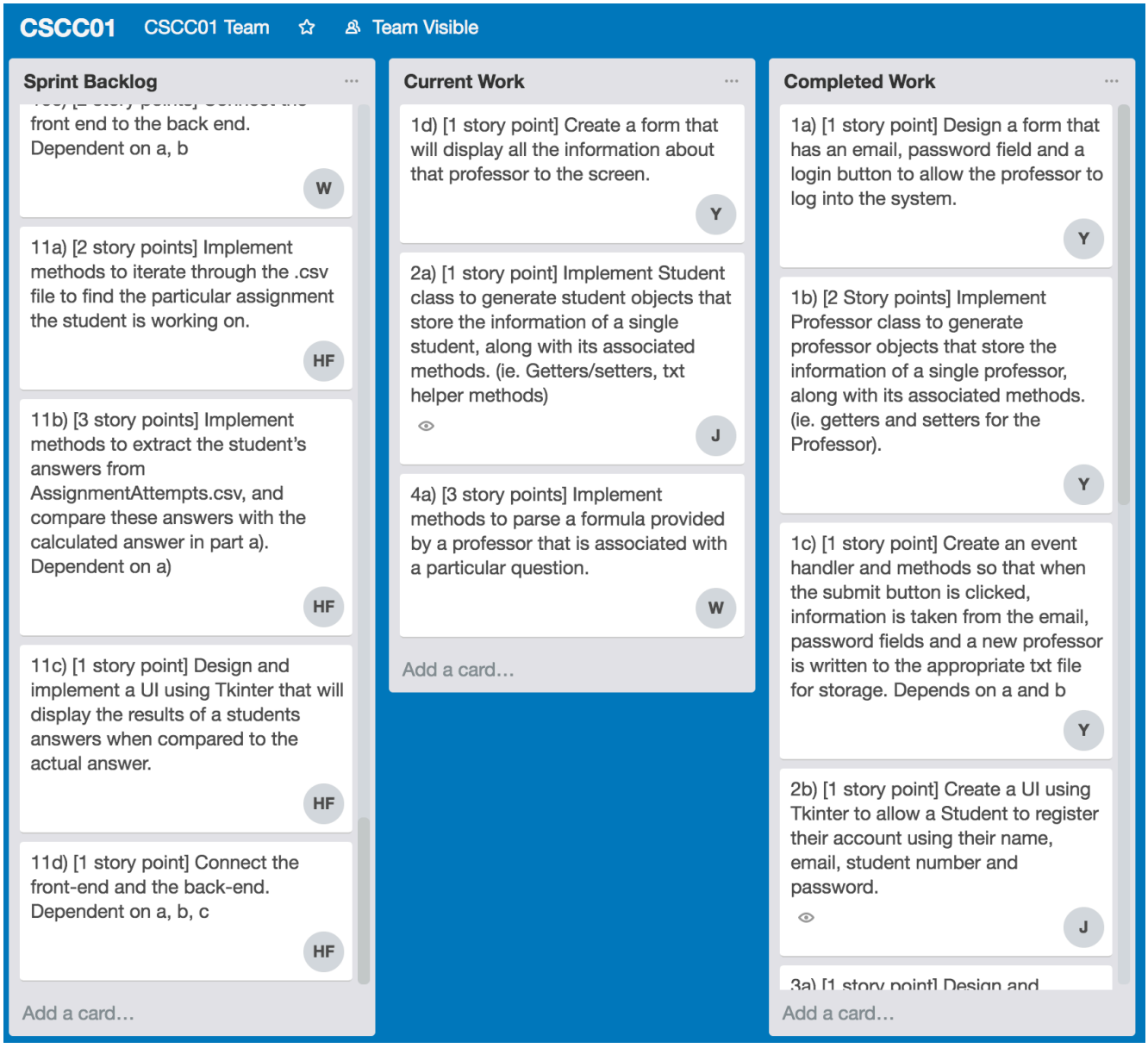
Start of Sprint

In addition to the screen shots from the end of Sprint 1, this is the taskboard including the tasks to be worked on during sprint 2.



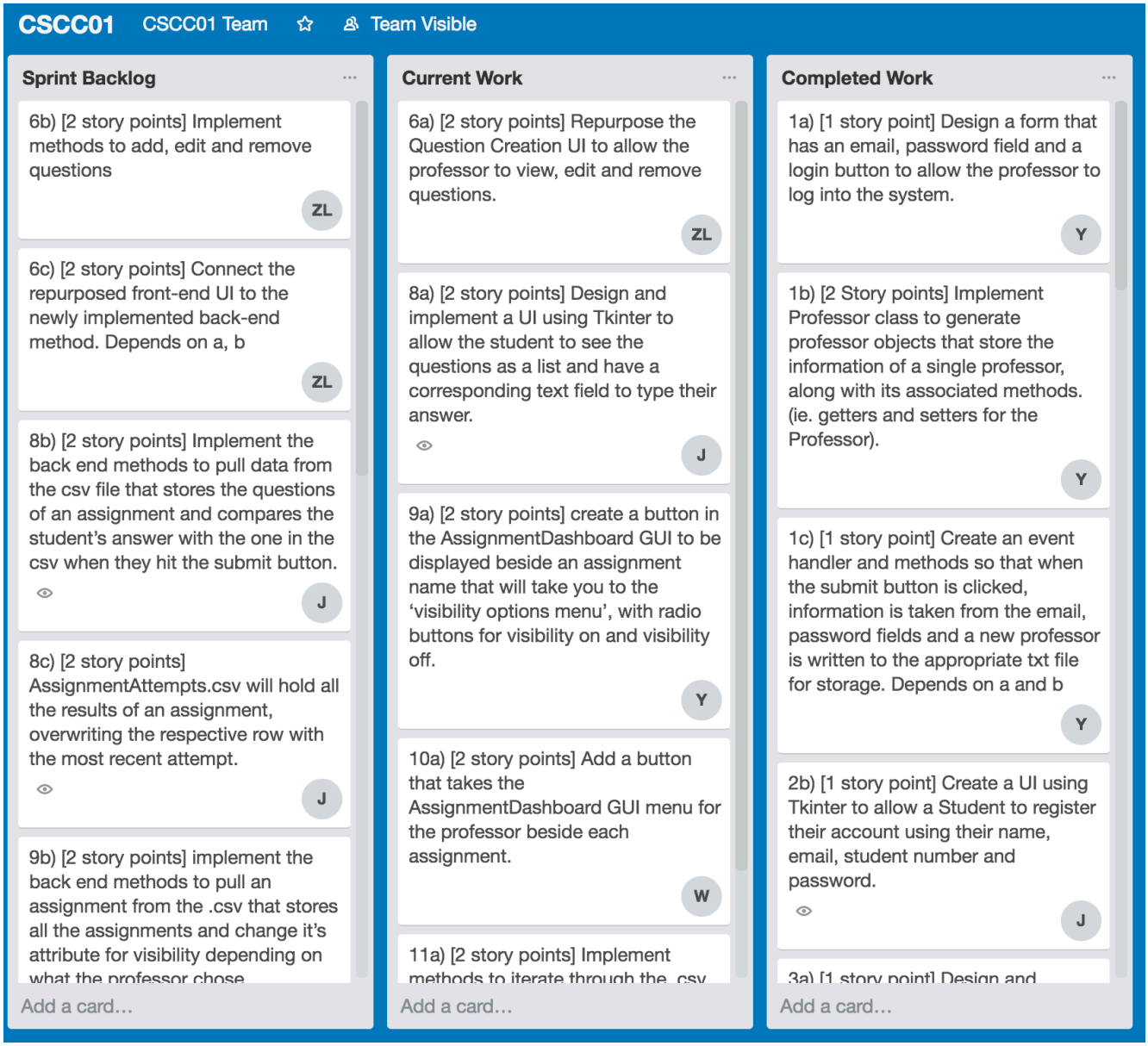

CSCC01
CSCC01 Team
Team Visible
Sprint Backlog
8c) [2 story points] AssignmentAttempts.csv will hold all the results of an assignment, overwriting the respective row with the most recent attempt.
J
9a) [2 story points] create a button in the AssignmentDashboard GUI to be displayed beside an assignment name that will take you to the 'visibility options menu', with radio buttons for visibility on and visibility off.
Y
9b) [2 story points] implement the back end methods to pull an assignment from the .csv that stores all the assignments and change it's attribute for visibility depending on what the professor chose.
Y
9c) [3 story points] connect the front and back end together. Dependent on a, b
Y
10a) [2 story points] Add a button that takes the
Add a card…
Current Work
1d) [1 story point] Create a form that will display all the information about that professor to the screen.
Y
2a) [1 story point] Implement Student class to generate student objects that store the information of a single student, along with its associated methods. (ie. Getters/setters, txt helper methods)
J
4a) [3 story points] Implement methods to parse a formula provided by a professor that is associated with a particular question.
W
Add a card…
Completed Work
1a) [1 story point] Design a form that has an email, password field and a login button to allow the professor to log into the system.
Y
1b) [2 Story points] Implement Professor class to generate professor objects that store the information of a single professor, along with its associated methods. (ie. getters and setters for the Professor).
Y
1c) [1 story point] Create an event handler and methods so that when the submit button is clicked, information is taken from the email, password fields and a new professor is written to the appropriate txt file for storage. Depends on a and b
Y
2b) [1 story point] Create a UI using Tkinter to allow a Student to register their account using their name, email, student number and password.
J
Add a card…

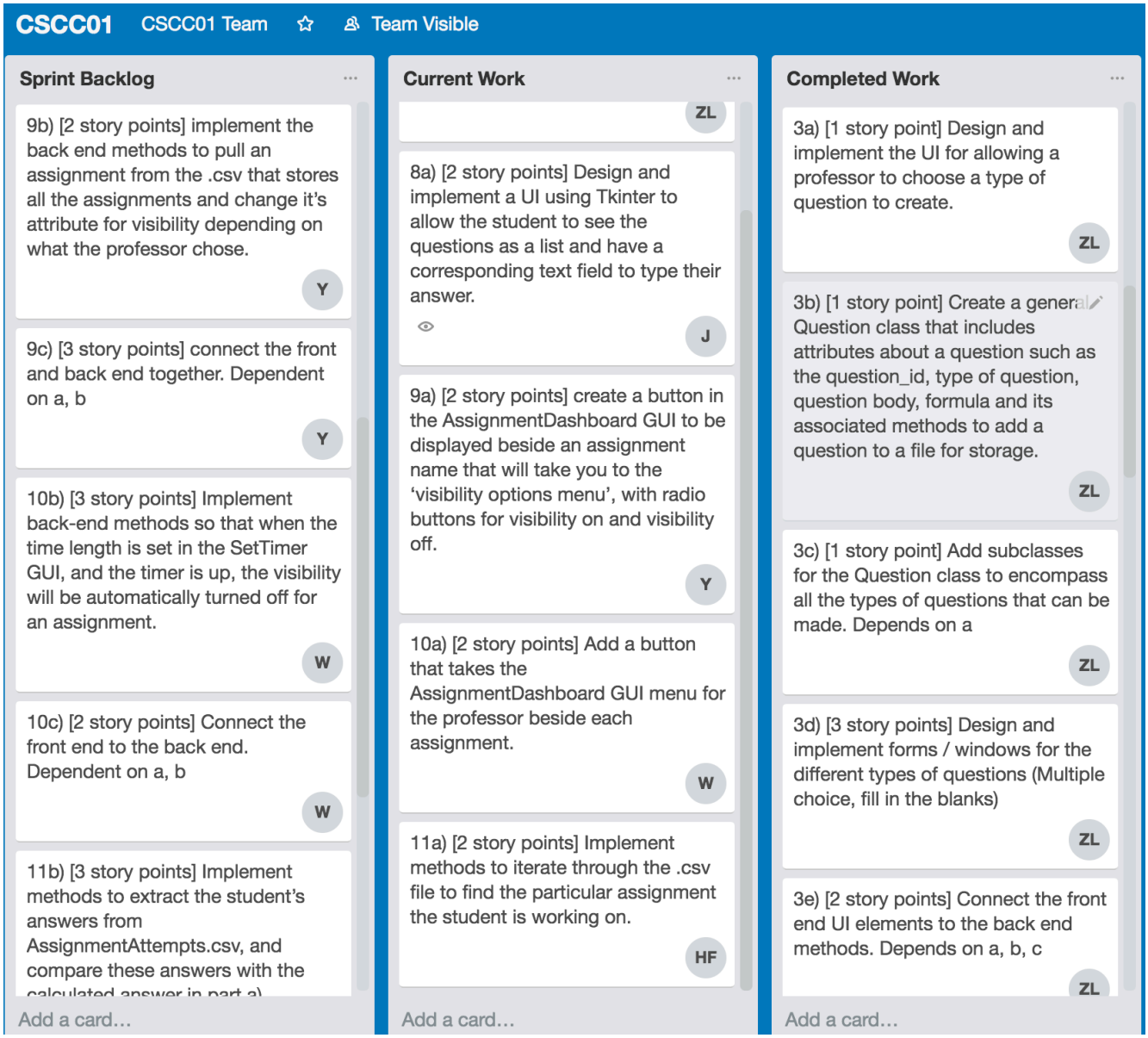

CSCC01
CSCC01 Team
Team Visible
Sprint Backlog
10a) [2 story points] Add a button that takes the AssignmentDashboard GUI menu for the professor beside each assignment.
W
10b) [3 story points] Implement back-end methods so that when the time length is set in the SetTimer GUI, and the timer is up, the visibility will be automatically turned off for an assignment.
W
10c) [2 story points] Connect the front end to the back end. Dependent on a, b
W
11a) [2 story points] Implement methods to iterate through the .csv file to find the particular assignment the student is working on.
HF
11b) [3 story points] Implement methods to extract the student's answers from AssignmentAttempts.csv, and compare these answers with the calculated answer in part a)
Add a card…
Current Work
1d) [1 story point] Create a form that will display all the information about that professor to the screen.
Y
2a) [1 story point] Implement Student class to generate student objects that store the information of a single student, along with its associated methods. (ie. Getters/setters, txt helper methods)
J
4a) [3 story points] Implement methods to parse a formula provided by a professor that is associated with a particular question.
W
Add a card…
Completed Work
1a) [1 story point] Design a form that has an email, password field and a login button to allow the professor to log into the system.
Y
1b) [2 Story points] Implement Professor class to generate professor objects that store the information of a single professor, along with its associated methods. (ie. getters and setters for the Professor).
Y
1c) [1 story point] Create an event handler and methods so that when the submit button is clicked, information is taken from the email, password fields and a new professor is written to the appropriate txt file for storage. Depends on a and b
Y
2b) [1 story point] Create a UI using Tkinter to allow a Student to register their account using their name, email, student number and password.
J
3a) [1 story point] Design and
Add a card…

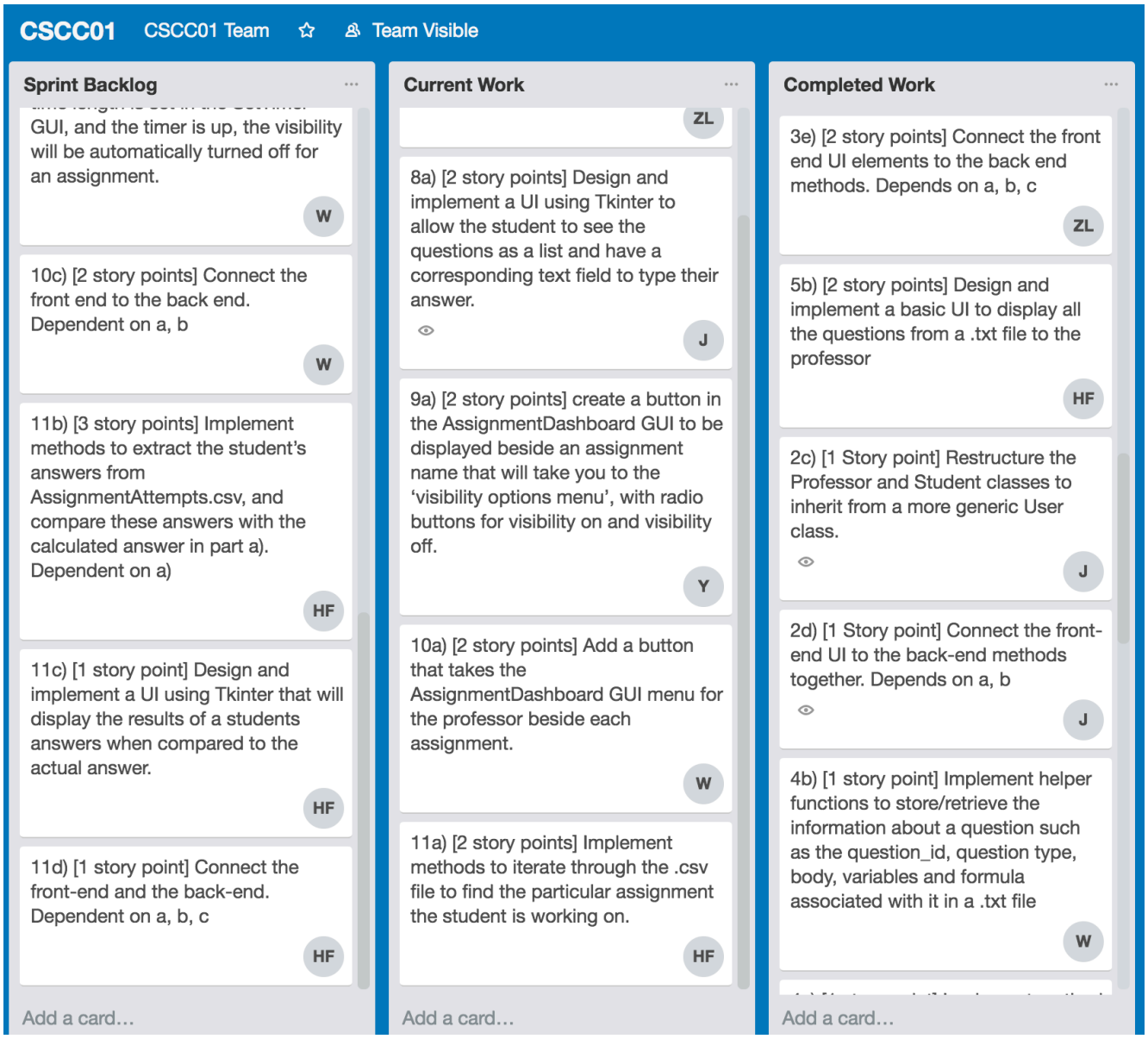

CSCC01
CSCC01 Team
Team Visible
Sprint Backlog
front end to the back end. Dependent on a, b
W
11a) [2 story points] Implement methods to iterate through the .csv file to find the particular assignment the student is working on.
HF
11b) [3 story points] Implement methods to extract the student's answers from AssignmentAttempts.csv, and compare these answers with the calculated answer in part a). Dependent on a)
HF
11c) [1 story point] Design and implement a UI using Tkinter that will display the results of a students answers when compared to the actual answer.
HF
11d) [1 story point] Connect the front-end and the back-end. Dependent on a, b, c
HF
Add a card…
Current Work
1d) [1 story point] Create a form that will display all the information about that professor to the screen.
Y
2a) [1 story point] Implement Student class to generate student objects that store the information of a single student, along with its associated methods. (ie. Getters/setters, txt helper methods)
J
4a) [3 story points] Implement methods to parse a formula provided by a professor that is associated with a particular question.
W
Add a card…
Completed Work
1a) [1 story point] Design a form that has an email, password field and a login button to allow the professor to log into the system.
Y
1b) [2 Story points] Implement Professor class to generate professor objects that store the information of a single professor, along with its associated methods. (ie. getters and setters for the Professor).
Y
1c) [1 story point] Create an event handler and methods so that when the submit button is clicked, information is taken from the email, password fields and a new professor is written to the appropriate txt file for storage. Depends on a and b
Y
2b) [1 story point] Create a UI using Tkinter to allow a Student to register their account using their name, email, student number and password.
J
Add a card…

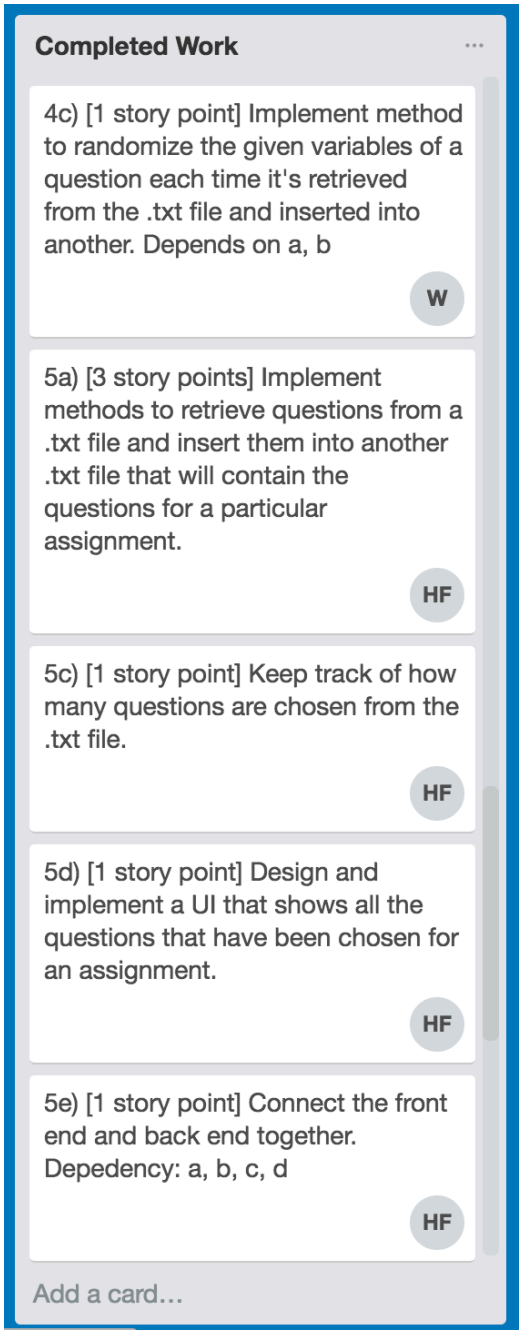
End of Sprint

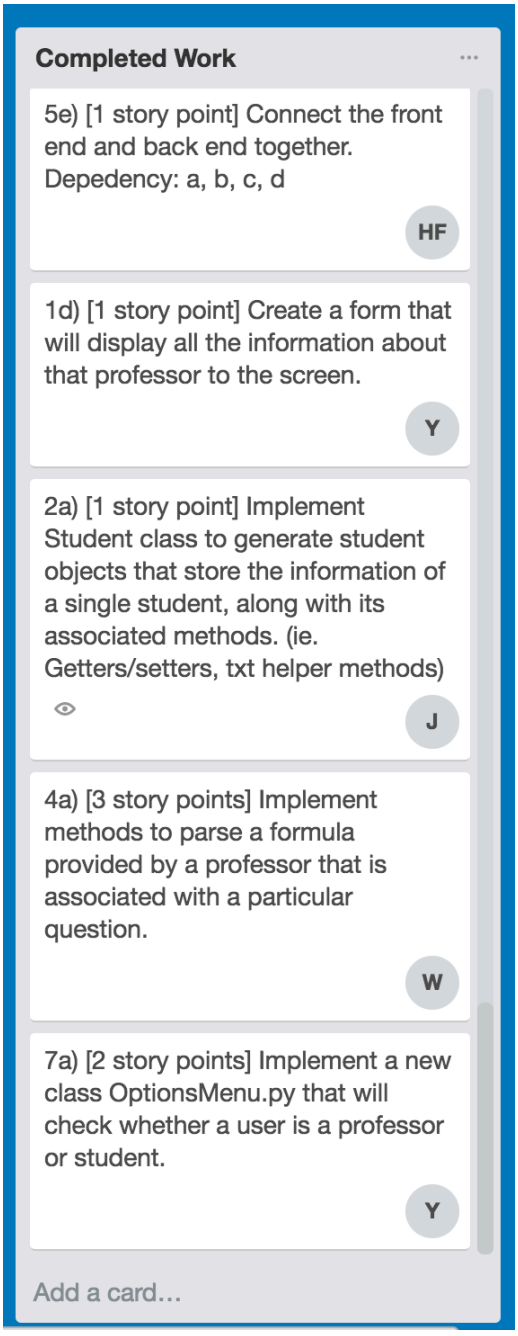
At the end of the sprint, all the unfinished tasks and user stories were completed and we began working on the tasks for the second sprint.




CSCC01
CSCC01 Team
Team Visible
Sprint Backlog
9b) [2 story points] implement the back end methods to pull an assignment from the .csv that stores all the assignments and change it's attribute for visibility depending on what the professor chose.
Y
9c) [3 story points] connect the front and back end together. Dependent on a, b
Y
10b) [3 story points] Implement back-end methods so that when the time length is set in the SetTimer GUI, and the timer is up, the visibility will be automatically turned off for an assignment.
W
10c) [2 story points] Connect the front end to the back end. Dependent on a, b
W
11b) [3 story points] Implement methods to extract the student's answers from AssignmentAttempts.csv, and compare these answers with the
Add a card…
Current Work
ZL
8a) [2 story points] Design and implement a UI using Tkinter to allow the student to see the questions as a list and have a corresponding text field to type their answer.
J
9a) [2 story points] create a button in the AssignmentDashboard GUI to be displayed beside an assignment name that will take you to the 'visibility options menu', with radio buttons for visibility on and visibility off.
Y
10a) [2 story points] Add a button that takes the AssignmentDashboard GUI menu for the professor beside each assignment.
W
11a) [2 story points] Implement methods to iterate through the .csv file to find the particular assignment the student is working on.
HF
Add a card…
Completed Work
3a) [1 story point] Design and implement the UI for allowing a professor to choose a type of question to create.
ZL
3b) [1 story point] Create a general Question class that includes attributes about a question such as the question_id, type of question, question body, formula and its associated methods to add a question to a file for storage.
ZL
3c) [1 story point] Add subclasses for the Question class to encompass all the types of questions that can be made. Depends on a
ZL
3d) [3 story points] Design and implement forms / windows for the different types of questions (Multiple choice, fill in the blanks)
ZL
3e) [2 story points] Connect the front end UI elements to the back end methods. Depends on a, b, c
ZL
Add a card…


CSCC01
CSCC01 Team
Team Visible
Sprint Backlog
GUI, and the timer is up, the visibility will be automatically turned off for an assignment.
W
10c) [2 story points] Connect the front end to the back end. Dependent on a, b
W
11b) [3 story points] Implement methods to extract the student's answers from AssignmentAttempts.csv, and compare these answers with the calculated answer in part a). Dependent on a)
HF
11c) [1 story point] Design and implement a UI using Tkinter that will display the results of a students answers when compared to the actual answer.
HF
11d) [1 story point] Connect the front-end and the back-end. Dependent on a, b, c
HF
Add a card…
Current Work
ZL
8a) [2 story points] Design and implement a UI using Tkinter to allow the student to see the questions as a list and have a corresponding text field to type their answer.
J
9a) [2 story points] create a button in the AssignmentDashboard GUI to be displayed beside an assignment name that will take you to the 'visibility options menu', with radio buttons for visibility on and visibility off.
Y
10a) [2 story points] Add a button that takes the AssignmentDashboard GUI menu for the professor beside each assignment.
W
11a) [2 story points] Implement methods to iterate through the .csv file to find the particular assignment the student is working on.
HF
Add a card…
Completed Work
3e) [2 story points] Connect the front end UI elements to the back end methods. Depends on a, b, c
ZL
5b) [2 story points] Design and implement a basic UI to display all the questions from a .txt file to the professor
HF
2c) [1 Story point] Restructure the Professor and Student classes to inherit from a more generic User class.
J
2d) [1 Story point] Connect the front-end UI to the back-end methods together. Depends on a, b
J
4b) [1 story point] Implement helper functions to store/retrieve the information about a question such as the question_id, question type, body, variables and formula associated with it in a .txt file
W
Add a card…


Completed Work
4c) [1 story point] Implement method to randomize the given variables of a question each time it's retrieved from the .txt file and inserted into another. Depends on a, b
W
5a) [3 story points] Implement methods to retrieve questions from a .txt file and insert them into another .txt file that will contain the questions for a particular assignment.
HF
5c) [1 story point] Keep track of how many questions are chosen from the .txt file.
HF
5d) [1 story point] Design and implement a UI that shows all the questions that have been chosen for an assignment.
HF
5e) [1 story point] Connect the front end and back end together. Depedency: a, b, c, d
HF
Add a card…


Completed Work
5e) [1 story point] Connect the front end and back end together. Depedency: a, b, c, d
HF
1d) [1 story point] Create a form that will display all the information about that professor to the screen.
Y
2a) [1 story point] Implement Student class to generate student objects that store the information of a single student, along with its associated methods. (ie. Getters/setters, txt helper methods)
J
4a) [3 story points] Implement methods to parse a formula provided by a professor that is associated with a particular question.
W
7a) [2 story points] Implement a new class OptionsMenu.py that will check whether a user is a professor or student.
Y
Add a card…