Personas

PROFESSORS

Karen Cohen

- 50-year-old Professor at UTSC, Female.
- Head of Statistics department.
- Stern personality, often cancels office hours.
- Only works on desktops at her home and university office; does not use a smartphone or laptop.
- Lectures STAB22, but her main focus is on her advanced machine learning research.
- Due to her busy schedule, she can only allocate time to teaching the course and nothing more.
- Before submitting final marks to the registrar’s office, she often asks her TAs to compile a spreadsheet of students’ marks at the end of the semester so she can review them.
- The difficulty of questions that she creates are based on averages of previous assignments.
- Loves the idea of working from home.
- Hobbies include learning new technologies to aid her research.

Ben Provinski

- 32-year-old Lecturer at UTSC, Male.
- Newly-recruited into the Statistics department.
- Lectures STAB22 and STAB27.
- Loves to engage with students.
- Professor Provinski often extends his office hours.
- Active on discussion boards and replies to emails fairly quickly (enjoys giving feedback to students).
- Likes to see statistics about his students, especially about their performance on quizzes and assignments.
- Appreciates feedback from students so he can actively adjust the course work.
- His laptop is his primary working machine and it is almost always on him.
- He’s comfortable doing work on his computer, but prefers grading on paper copies of assignments.

- Does remarking for his courses himself.

TEACHING ASSISTANTS

Jenny Li

- 20-year-old TA at UTSC, female.
- 3rd year Statistics specialist.
- TA for STAB22.
- Currently taking 6 courses, each with an assignment or quiz every week meaning time is severely constrained.
- Since her bag is always full of school material, she prefers when her students' remark requests can be handled digitally.
- She lives 1.5 hours away from UTSC and because of her long commute, she is always on her Android phone often checking emails.
- Prefers to complete her work on the go using her Windows tablet.
- When she has time, she uses her laptop for light browsing, school work and entering student quiz marks.
- Comfortable with computers; has experience with document typesetting in LaTeX and programming with R.

STUDENTS

Dana Fath

- 19-year-old student at UTSC, Female.
- Second-year student majoring in Life Sciences.
- Uses a computer for social networking, writing lecture notes, and completing assignments.
- Reads lecture notes on her phone while riding the TTC.
- Familiar with most common web applications and software.
- Comfortable being introduced to new software.
- Completes assignments gradually instead of finishing them in one session.
- Prefers assignments with multiple choice questions because there are less grading errors.
- Always requests detailed feedback from professors on grades.
- Often disputes her marks.

User Stories

- ❑ **Priority 1 (Creating problem sets with a deadline)**

U1: As Karen/Ben (a statistics Professor) and Jenny (a TA), I would like to be able to create a new assignment and write problems with their corresponding solutions and multiple choice options.

- ➔ **T1:** Build functionality in Java for a user to create a new assignment and enter problems and solutions through the command-line in which the assignment gets saved as a .csv file (assignment#.csv, where # is the assignment number) in the format:

  ```
  problem id (integer), problem (string), solution (string),
  optionA|optionB|optionC|optionD (string)
  ```

  Where each problem is on its own line and the first row in the .csv file is dedicated to information regarding when the assignment was created and when it is due.

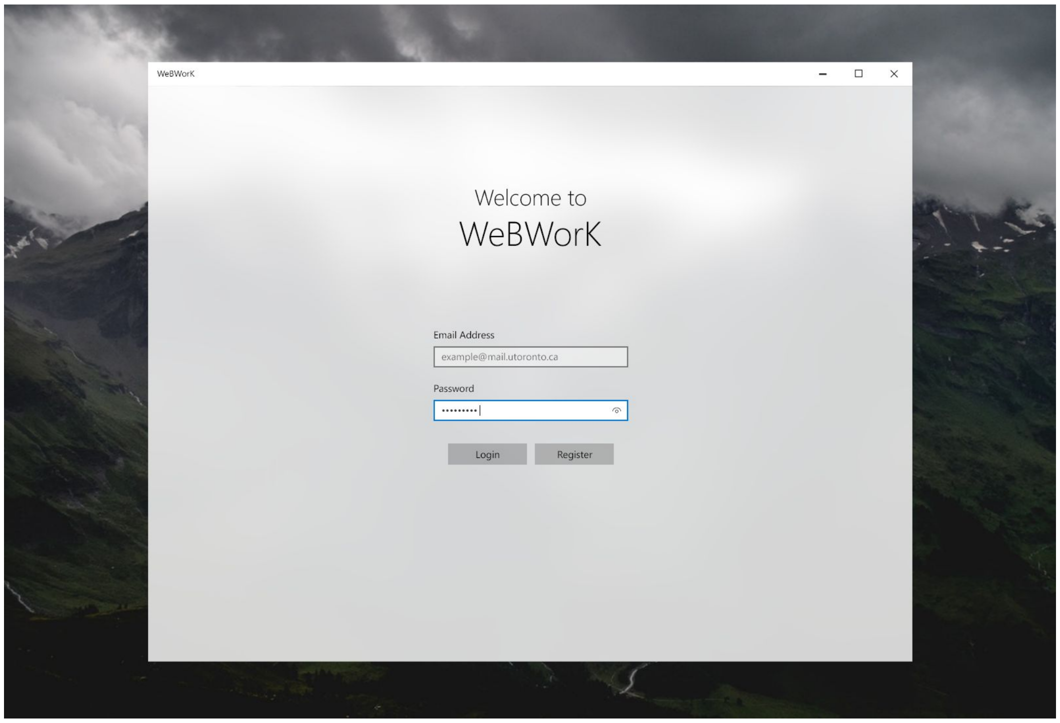
- ➔ **T2:** Add functionality to register and login from command-line by storing/loading users and their hashed passwords located in a .csv file (users.csv) in the format:

  ```
  user id (integer), hashed password (string)
  ```

  When user registers, hash their password using SHA256 and upon logging in, hash the password the enter with the one in users.csv to determine a match.

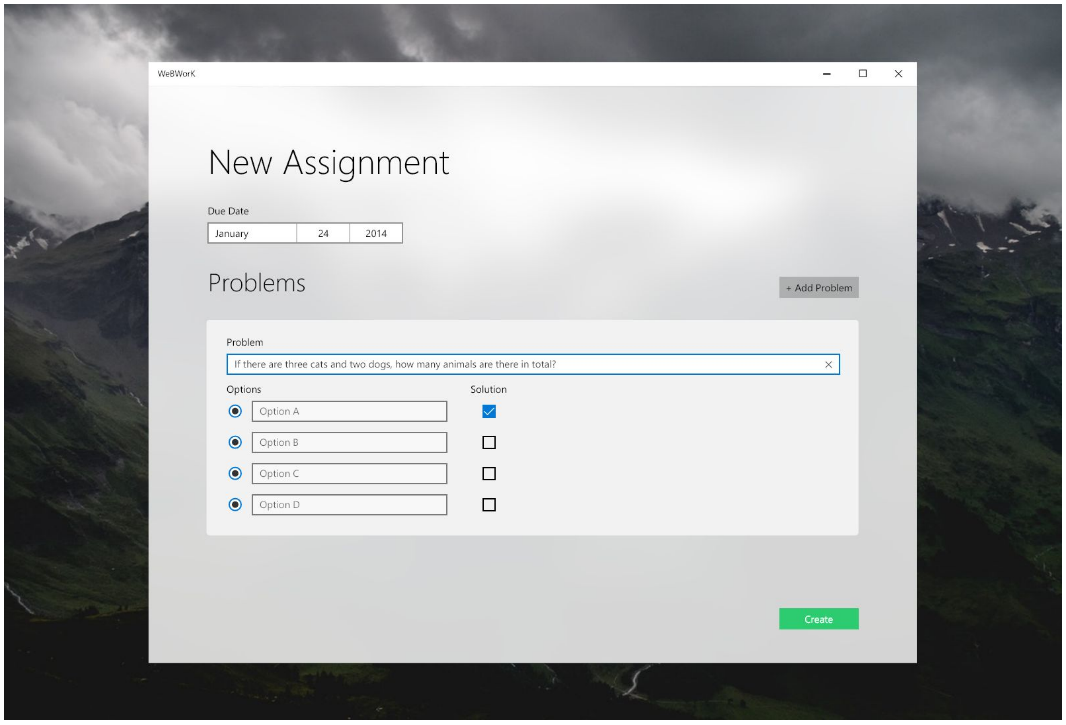
- ➔ **T3:** Develop registration and login UI (UI window 1) with Swing and integrate existing command-line functionality.

Mockup:



➔ **T4:** Build interface (UI window 2) using Swing/AWT for TAs and Instructors to with functionality of the command-line version for creating assignments.

Mockup:



Due Date: January 24 2014

Problem: If there are three cats and two dogs, how many animals are there in total?

U2: As Karen/Ben (a statistics professor) and Jenny (a TA), I would like to be able to modify or remove an existing assignment question and/or solution.

➔ **T5:** Implement UI (UI window 3) that lists all assignments that are released/unreleased by parsing through all assignment files in the directory

➔ **T6:** Create button per assignment listing to edit the corresponding assignment problems. The button ultimately takes the user to the same (or slightly modified) assignment creation UI (UI window 2), in which the assignment's data is retrieved from the file (assignment#.csv), but with "Save" at the bottom instead of "Create".

U3: As Karen/Ben (a Statistics professor), I would like to release problem sets to my students and set deadlines for these assignments.

➔ **T7:** Modify assignment creation java file (AssignmentCreator.java) to include "unreleased" flag in first row of .csv file

```
creation date (dd/mm/yyyy), due date (dd/mm/yyyy),
            "unreleased"/"released"
```

➔ **T8:** Implement a "release" button on each unreleased assignment listing such that when it is pressed, the released flag in the csv file (assignment#.csv) is changed to true

U4: As Karen/Ben (a statistics professor) and Jenny (a TA), I would like for every user to be asked for authentication before accessing their assignments, to differentiate between instructors and students and stop students from editing/creating assignments.

- ➔ **T9:** Create a main file that instantiates the login/registration GUI.

- ➔ **T10:** Edit the format of (users.csv) to include a new parameter: instructor/student.

  is_instructor (boolean), user id (integer), hashed password (string)

  - ◆ When a user register a new account, prompt a window asking whether they are a student or instructor.

- ➔ **T11:** Edit the register method in Authenticator.java, to include the Is instructor flag at registration.

- ➔ **T12:** If login was successful, open the assignment listing window according to the user type.

❑ **Priority 2 (Solving problem sets)**

U5: As Dana (student), I would like to view all my open and in-progress assignments along with their due dates.

- ➔ **T13:** Create a Swing interface for students to see a list of "released" assignments by parsing through each assignment#.csv file and only displaying the ones that have the released flag set to true

U6: As Dana (student), I would like to be able to select an open problem set before its deadline has passed and begin answering questions.

- ➔ **T14:** Create an "assignment interface" in which questions are randomly selected from the assignment .csv file (assignment#.csv) and displayed with their options as radio buttons.

U7: As Dana (student), I would like save my progress on my current assignment so I can work on it later.

➔ **T15:** Create a "save progress" button on the assignment interface that adds the student's attempted answers into a .csv file (assignment#_submissions.csv) for that assignment that contains one student per row. The format is as follows:
```
Student id (integer), q1 answer (integer 1-4), … qn answer, average
   mark (integer 0-100), # tries (integer), time spent in seconds
              (integer), final mark (integer 0-100)
```
Where n is the number of questions for that assignment.

U8: As Dana (student), I would like to be able to submit my completed assignment before the deadline has passed.

➔ **T16:** Create a "submit" button that updates the student's answers in the submissions file (assignment#_submissions.csv) to reflect their final submission and increments the number of tries.

❑ **Priority 3 (Auto-marking & student feedback)**

U9: As Karen/Ben (a statistics professor), I want every assignment submission to be auto-graded by comparing the solutions I made for the assignment to student solutions.

➔ **T17:** Create a function to parse through the submissions file (assignment#_submissions.csv) and compare each student's solutions to the one in the assignment file (assignment#.csv) and saving their final mark in the submission file.

U10: As Dana (a student), I would like to see which questions on an assignment I got right and wrong after a submission so that I can retry the assignment if the deadline has not passed.

➔ **T18:** Create a "submission summary" interface that is shown after a student hits the "submit" button which displays all the questions and a check mark or cross denoting whether it was answered correctly or not.

U11: As Karen/Ben (a statistics professor), I want to be able to view a student's grades for all closed assignments that they have submitted.

➔ **T19:** Create a basic command-line interface for an instructor to enter a student's id and print all assignments with their corresponding grades by parsing through each of the submissions files (assignment#_submissions.csv) and locating the student's entry

➔ **T20:** Create a GUI for displaying the student's assignment marks

U12: As Dana (a student), I would like to receive detailed feedback about my assignment submission after the deadline has passed regarding how much time I spent on the assignment, how many tries I took, and the average mark of all my completed attempts.

➔ **T21:** Create button on the interface that lists all completed/closed assignments beside each listing to return the logged-in student's time spent, tries attempted and average mark of all attempts from the (assignment#_submissions.csv).

❏ **Priority 4 (Grade analysis & grade retrieval)**

U13: As Ben (a statistics professor), I would like to see the grade distribution of an assignment over the student body after the deadline has passed.

➔ **T22:** Create function to compute average of an assignment by parsing through the submissions file (assignment#_submissions.csv) and using the final mark column to compute the mean

➔ **T23:** Add the text UI element to display the course average beside the assignment listing

U14: As Karen (a statistics professor), I would like the application to generate a spreadsheet that I can download of all student marks for a specific assignment.

➔ **T24:** Implement UI that lists all assignments that are closed by parsing through all assignment files in the directory

➔ **T25:** Create button beside a closed instructor's assignment listing to export the student's marks to a location the instructor desires (essentially a 'save as' menu that saves the (assignment#_submissions.csv for that particular assignment to the location the instructor chooses)

❏ **Priority 5 (Post Auto-Mark; remarking)**

U15: As Dana (a student), I would like to be able to submit a remark request with an explanation after the assignment deadline has passed so that my grade can be reevaluated.

➔ **T26:** Lists student assignments based on whether they are closed or open

➔ **T27:** Create a "Remark" button on the assignment listing, and enable it after the deadline for the assignment has passed. When pressed, a message box is prompted to the screen and the message is sent to the Instructor's email.

Removed! [remark requests go directly to TA's email, can be printed that way]

~~**U16:** As Jenny (a TA), I would like to be able to print a student's remark request that includes their version of the assignment, their answers, and their justification paragraph so that I can look over the request offline.~~

U16: As Jenny (a TA), I would like to be able to manually adjust a student's assignment submission mark after they have submitted a remark request.

➔ **T28:** Create an interface with text fields for a student number, assignment #, and new grade with a save button that saves to the assignment#_submissions.csv file for that specific student. [Also create button on instructor listing screen to take the instructor to this new interface]

U17: As Dana (student), I would like to be able to be notified via email when my remark request has been processed with information regarding the instructor's decision.

➔ **T29:** When the "Remark" button is pressed on the remark interface, send an email with the assignment # and final grade to the student.

❑ **Priority 6 (TA groups for remarking)**

Removed!
[Removed due to the need of prioritizing implementation of fundamental requirements]

~~**U18:** As Karen (a Statistics professor), I would like to assign groups of students to TAs so that every TA is in charge of dealing with remarks for only a section of the student body.~~

Removed! [Removed since there are are no longer TA sections]

~~**U19:** As Jenny (a TA), I would like to be notified via email when a student that is of a section assigned to me submits a remark request so that I can look over their submission.~~

Removed! [Removed since all instructors will receive remark requests]

~~**U20:** As Ben (a Statistics professor), I would like the option to receive student remark requests directly instead of going through TAs so that I can remark assignments myself.~~

- ❑ **Priority 7 (Course feedback)**

U18: As Dana (a student), I would like to be able to write feedback for the professor and TAs about an assignment after the deadline has passed.

- ➔ **T30:** Add a feedback button to the right corner of the student listing.
- ➔ **T31:** Implement a GUI that enables student users to write a feedback text anonymously and write it to the file studentFeedback.csv. Format for the file where every row is a student feedback:

```
Subject, message, date (dd/mm/yyyy)
```

U19: As Ben (a Statistics professor), I would like to be able to see comments from students about a problem set after the deadline has passed, so I can gauge difficulty and potentially adjust future assignments.

- ➔ **T32:** Create a Feedback class to contain the time and feedback.
- ➔ **T33:** Create method to extract all student feedbacks for a given assignment and store them into Feedback objects.
- ➔ **T34:** Create a button beside each closed assignment in the instructor listing GUI to show all the feedbacks for that given assignment (need to create a new GUI to display the feedbacks).