Documentation for HCI Project

1 Use cases

This section contains our 4 use cases.

1.1 File Submission

- **Agents:**
    1. Student
- **Purpose:** Grading platforms should be capable of collecting assignment submissions, timestamped and stored in a central location
- **Procedure:**
    1. Student visits authenticated landing page, which shows outstanding assignments for courses
    2. Clicking on the assignment goes to the assignment page, containing the project spec, an accessor for the most recent previously submitted file (if applicable), and a submission interface
    3. If a small number of files is to be submitted, simply dragging and dropping/upload dialog for each slot will work
    4. If a larger/unknown number of files is to be submitted, uploading a zip file will happen instead. If enabled, jGrader will unzip it and attempt to match the contents to any file-specific slots that may exist.
    5. After uploading, the file submission slot will turn grey and show a confirmation message until the page is refreshed. (After refreshing, the previously submitted accessor will be shown)

1.2 Excuse Request[1]

- **Agents:**
    1. Students
    2. Professor

[1]This use case is only partially implemented in the prototype.

3. Registrar (not directly)

- **Purpose:** Request being officially excused for a task
- **Procedure:**

1. Student visits authenticated landing page, which shows outstanding assignments for courses.
2. Clicking on an assignment navigates the student to the assignment page which (among other things) has a button to request an excuse for the specific assignment.
3. The student clicks a button to request an excuse for this task from the professor.
4. The task is marked as "EXCUSE PENDING" on the course page. [2]
5. The professor is notified of the request via an email that is sent to them. This email includes a link to a page on Grader where he/she can confirm the extension request.
6. (not directly in the system) the student sends a medical excuse to the registrar and waits for them to confirm it (to the professor).
7. The professor clicks on the link in the mail that was sent to him/her earlier
8. The professor lands on an (authenticated) page on jGrader where he/she can either confirm or deny the requests.
   - If the excuse was accepted by the registrar, the professor clicks the "Grant Excuse" button
   - If not, he/she clicks the "Deny Excuse" button
9. The student is notified via email if the excuse has been granted or denied.
10. At the same time the excuse is marked appropriately on the landing page.

[2] This and the following steps are not available in the prototype.

1.3 Extension Request[3]

- **Agents:**
  1. Students
  2. Professor
  3. Registrar (optional, indirectly)
- **Purpose:** Student has an emergency and is not able to complete the assignment at time.
- **Procedure:**
  1. He logs in to jGrader and sees the current timeline of tasks.
  2. He clicks on the assignment link which has a button to request an extension for the current assignment.
  3. He selects from a calendar view the available date when he can submit the solution and complete a message to motivate the extension.
  4. The website shows him a confirmation that an email has been sent to the professor regarding the extension.
  5. The professor receives an email where he has 2 options either to approve/deny the extension request.
  6. The professor decides on approving/not approving and the student receives an email regarding the new time.

     Confirmation of an illness from the registrar might be required
  7. The timeline of the student is changed accordingly.

1.4 Grading

- **Agents:**
  1. Student(s)
  2. Teaching Assistant/Instructor

[3]This use case is not implemented in the prototype.

- **Purpose:** A student has submitted assignments, and the grades for the work must be entered into the gradebook.
- **Procedure:**
    1. Teaching Assistant logs into jGrader and selects the course they are grading for from a list
    2. The TA selects the assignment/task submission to be graded
    3. After loading the page, the TA either downloads all submissions, or they download an individual's submission by clicking the down arrow next to the person's name
    4. The TA opens each submission and grades it accordingly on their local system outside of jGrader
    5. The TA enters their feedback into the feedback field, and the grade into the grade field, and clicks update.

        Alternately, they can skip the updating and just update all at once by clicking "Update All"
    6. The grading dashboard now shows the current grading status for the assignment, which is hopefully 100% graded.

2 Design Documentation

In this section you can find design documentation about individual pages of the system. The title of each section reflects only who wrote the respective section, not who designed and implemented the page in the prototype.

2.1 Authentication Page (Tom)

The authentication page is the entry point to the system and as such we designed it very carefully. It exists in 3 versions:

1. a normal version,
2. a version for when the user enters an incorrect password and
3. a version for when the user has just signed out of the system.

Each of the versions has the same basic layout. The only real content of the page is centered. It contains a heading, input elements for username and password as well as a submit button. We designed it this way because we want to draw the users attention to its use, being as minimalistic as possible. However because it is the first contact any user has with the system we wanted to leave an impression and added a fancy[1] background image. EdN:1

When the user accesses the sign in page for the first time, the keyboard automatically focuses the username field. This makes it easy for the user to sign into the system. When the user enters a wrong password they end up the second version of the page, which in addition has an error message and a forgot password link. The keyboard is automatically focused on the password field so the user can quickly re-enter the password since it is more likely that the password, and not the username, is incorrect. To further speed up loggin in, the username field keeps the value entered in the first sign in attempt.

During (paper) prototype testing a tester mentioned that it would be convenient to have reset password functionality, so we added a link for it. On the linked page the user can enter their username and then gets an email with a link to reset the password[4]. This page does not have the username pre-entered or an auto-focus functionality because we want to prevent abuse of the system.

The third version of the authentication is shown after the user signs out of the system. It contains a small message to ensure that the user knows they have been signed out. Furthermore if auto-focuses the username field to enable the user to quickly sign in again. An earlier version did not contain a sign in form but only a message that the user had been logged out, however this was criticized during paper prototype testing.

2.2 Student Landing Page (Naomi)

- Students are provided with an overview of upcoming deadlines and how much of the tasks are done (e.g. if the student finished 4 out of 5 tasks it will visually show 80%). This will help students remember all their deadlines.

[1] EdNote: Replace this with a different word?

[4] Because this is simply a prototype only the form is implemented, not the actual mail being sent. We also do not have a layout for the reset password page. If this system were to be used at Jacobs, the reset link might be removed because CampusNet accounts are used and we can not reset the password externally.

- Links on the visual representation of the percentages will allow students to quickly navigate to the assignments that are due.
- To limit the information that is on the screen previous courses are only shown when the student clicks on the rider for previous courses. This ensures that the page provides an easily processable overview without clutter.
- In the Enrolled Courses section, the student can access all courses he is currently enrolled in.

2.3 TA Landing Page (Nick)

The TA landing page is the same as the stuent one. In this demo we did not want to duplicate all the menu items for the TA, we have not added any content. In an actual implementation of the system, the menu entry of the TA landing page would include the TAs courses under "Current courses" and "Previous courses" respectively.

2.4 Profile Pages (Tom)

[2] EdN:2

2.5 Current courses page (Roxana)

[3] EdN:3

2.6 Previous courses page (Vlad)

[4] EdN:4

2.7 TA Grading Page (Nick)

- Assignments, sorted in descending order by ‘’% ungraded’‘, followed by ‘’most distant due date in the past’‘ are selected from a main TA

[2] EdNote: Write this section
[3] EdNote: Write this
[4] EdNote: Write this

dashboard for the course showing numbers of graded tasks, a list of students, and other information.

- ''We placed the comment box before the grade box in order to encourage TAs to provide comments for grades; when tabbing through the page, the comment field will be selected first'‘

- Ample space is provided between assignment entries in order to reduce the chance of visual confusion. In a complete implementation, JavaScript would also be used to turn the currently selected row/row fields are in grey in order to further differentiate what's being edited.

- A single submit grades button, which floats throughout the page for easy access, reduces the number of steps necessary to enter a grade, which is important for TAs when grading large numbers of small assignments.

- Different sizes between the comment and grade fields provide further visual differentiation.