Case Study

Draw a use case diagram of the blackboard system described as below:

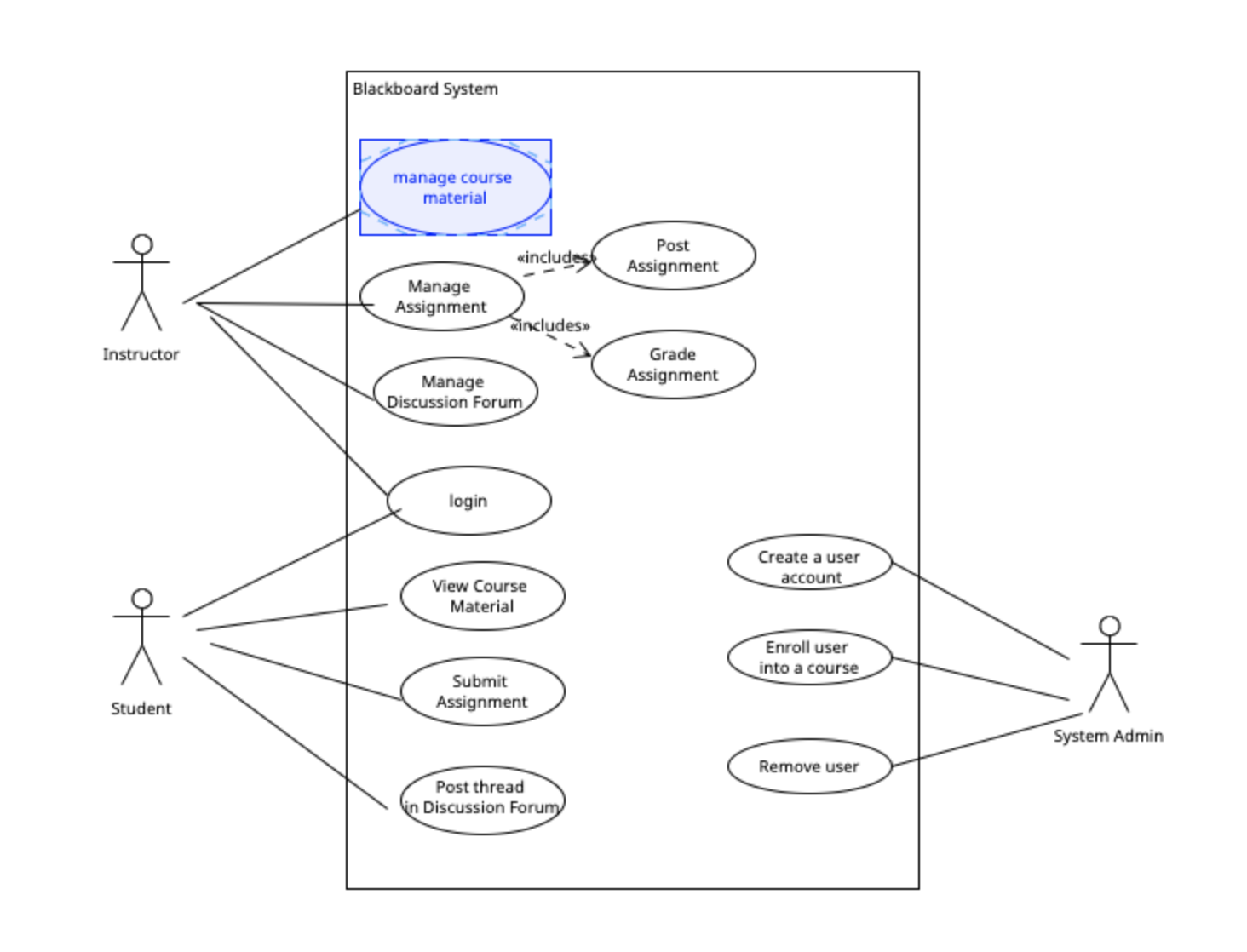
1. A new BU student, Bob registered the software engineering course CS673 taught by Prof. Zhang.
2. System admin created an account on the blackboard for Bob and enrolled him into CS673 class.
3. Bob can log into the blackboard and view the CS673 course site.
4. Prof. Zhang logged into the blackboard. She posted the syllabus on blackboard, as well as an announcement to welcome students before the first class. The announcement was automatically sent to all students through email.
5. Bob received the email and logged into the blackboard to check the syllabus.
6. Prof. Zhang posted the lecture slides and related course materials on the blackboard before the class.
7. Bob previewed the lecture content before the class.
8. Prof Zhang updated some materials after the class, and also posted an assignment on the blackboard.
9. Prof. Zhang created a discussion forum on blackboard to facilitate students' communication.
10. Bob reviewed the course material. He also posted a question on a discussion forum about the use case model. His classmate Ann posted her answer to that question. Some other students also joined the discussion.
11. The discussion helped Bob understand the use case model. He completed the assignment and submitted it just before the deadline.
12. Prof. Zhang graded the students' submissions.
13. Bob reviewed his grade of the assignment and feedback from Prof. Zhang.
14. Unfortunately, he had some work schedules changed and had to drop the class.
15. The system admin deleted his name from the course list.
16. Bob cannot view the CS673 course website anymore.
17. Identify actors and use cases. Draw a use case diagram.

Actors:

* Admin
* Teacher (Prof. Zhang)
* Student (Bob + Anne)

Use cases:

1. The blackboard system allows students to register for a particular class(?)
2. Admin add course to Blackboard ?
3. Admin created a ‘professor’ user/account for a new professor ?
4. Professor post material to blackboard
5. Professor can grade user submissions
6. Student submit assignment
7. Student post thread in the discussion forum
8. Admin can add/remove users to/from classes
9. Students drop class(?)

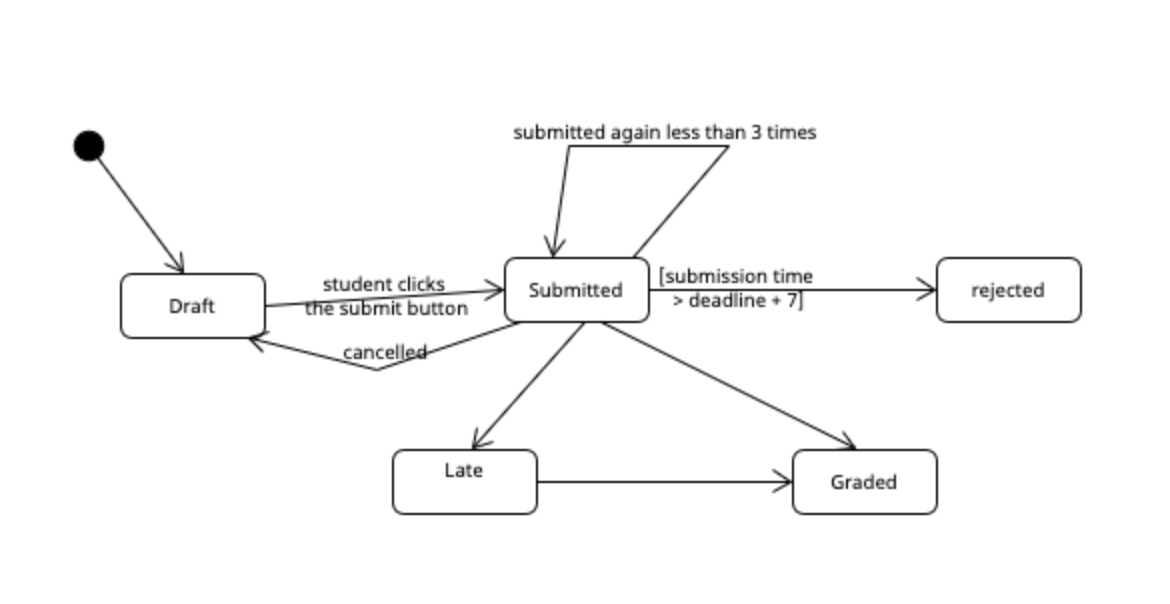


1. Identify entity classes

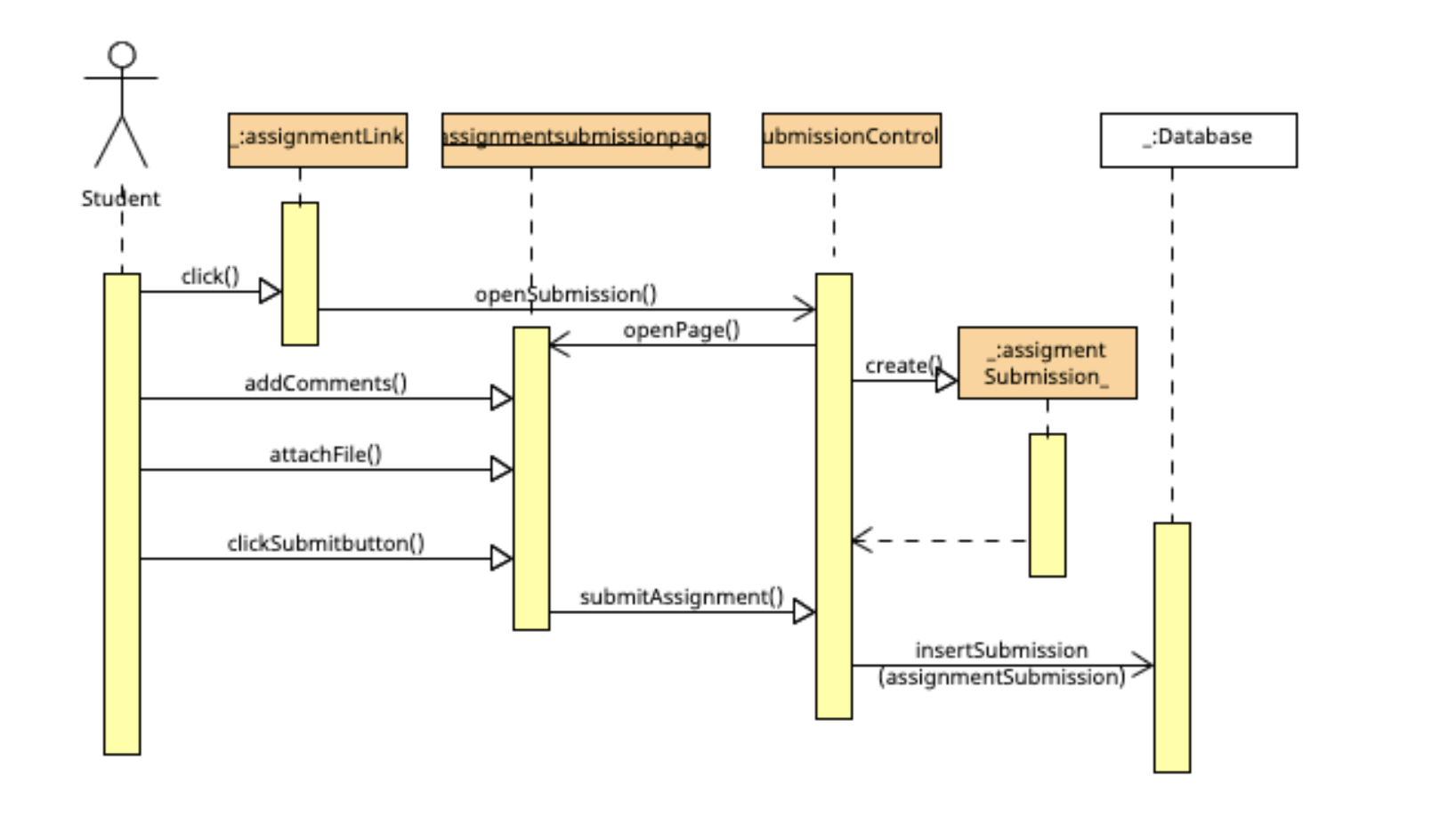
On the CS673 blackboard course site, students can submit their completed assignment by clicking on the corresponding assignment. A new *page* will be displayed. Students can upload their files, write the comments, and click the *Submit button* to **submit their assignments**. A student can also first save the submission and submit later, as in our blackboard system*.*  Each student can submit up to 3 times, but only the last submission will be counted. Each assignment has a deadline. The late submissions will be penalized within a week with 3% of your grade each day. No assignments will get credits one week after the deadline. After the deadline, the instructor will grade each student’s submission by clicking on the corresponding submission and assigning a score. The instructor can also add comments or attach additional files as feedback for students.

1. Identify entity, boundary, and control classes
2. Please draw a state transition diagram to **show the status of the single AssignmentSubmission object.**

States: Created - > Submitted , rejected , late -> Graded,



1. Draw a sequence diagram to show the object interactions in the above description related to assignment submission and grading.



1. Apply risk-based domain testing techniques (particularly equivalence class and boundary value analysis) to design test cases for the above use case “submit assignment”. Please state your assumption if there is anything ambiguous. This will include:
   1. Identify the domains(variables) involved: files, comments, submit button, submission date, submission times,
   2. Identify the relevant dimensions of each variable. Files:( name (length, character), type, size, location), comments (length, character),
   3. Apply equivalence class and boundary value analysis to each variable and variable combinations to design test cases.

| Variable | Dimension | Equivalence class | Best representative | Risks | Notes |
| --- | --- | --- | --- | --- | --- |
| File | File Type | Allowed document/ picture files | Txt, pdf, doc, jpg, | Mishandling legitimate file types |  |
|  |  | Executable file types, or unknown file type | .exe, .lib, random binary file. | May cause malicious code to be loaded into the server |  |
|  | File size | Allowed normal file size | 1byte, 1MB | Mishandling legitimate file size |  |
|  |  | Too small or too large file | 0 byte,  2MB  1GB | Meaningless file  Wrong file size may overload the server. |  |
|  | File Name Length |  |  |  |  |
|  | File Name Characters |  |  |  |  |