InstructED

Software Test Plan

CSCI-P465/565 (Software Engineering I)

Project Team

Kevin Cao

Andrew Jedlicka

Blake Miller

Shaun Trimm

1. Overview (Sprint 1)
   1. Test Objectives
      * Verify a user can register an account
      * Verify a user can log into their account using their email and correct password
      * Verify a user can reset their password using a password reset link sent to the email tied to their account.
      * The password a user submits is encrypted
      * Verify a user can log using Facebook Oauth.
      * Verify Duo 2 Factor Authentication works, and is required to log in.
   2. Test Environment
      * 1: Local testing on a Linux system (Arch Linux), using the Brave browser (Chromium) and a locally hosted Postgresql database
      * 2: Non-Local testing using a Heroku based web server, and Postgresql database server
   3. Test Personnel
      * Kevin Cao
   4. Acceptance Criteria
      * A user can register and login, as well as reset their password. All links and buttons direct to where they are supposed to and no pages that can only be accessed by users can be accessed by non-users.
   5. Noted Omissions
2. Test Cases (Sprint 1)
   1. Test Case 1: Normal Login
      * Description: A user with an account would like to log into their InstructED account
      * Initial Conditions: User must already have an account, and know their password and email address
      * Input Data: Email of the account, password for the account, and phone number (for Duo Authentication)
      * Specifications: User can log into their account with the correct email and password, and Duo 2FA works
      * Procedure: Using an account created using one of our emails, a password we know, and our phone number we would test the login procedure
   2. Test Case 2: Forgot Password
      * Description: A user has forgotten their password, and wants to reset it
      * Initial Conditions: User must already have an account, know their email address and have access to the email address tied to their account.
      * Input Data: Email of the account, and the new password.
      * Specifications: A user can reset their password using a password reset link sent to the email tied to their account
      * Procedure: We would use the email associated with a test account to reset the password of the test account, then attempt to login with the new password.
   3. Test Case 3: Registration
      * Description: A new user would like to create an InstructED account.
      * Initial Conditions: New user must have an email address to use for their account, and a password to provide.
      * Input Data: Email of the new user, first name, last name, and password
      * Specifications: User can log into their account with the correct email and password.
      * Procedure: We would use one of our emails, and a password to register for an account, then verify it exists by checking the database and logging in.
   4. Unsuccessful Login
      * Description: A user would like to log in, but either types the wrong email address or wrong password
      * Initial Conditions: User must have an account
      * Input Data: Erroneous password/email address inputted
      * Specifications: User can log in with their correct email address and their correct password
      * Procedure:
   5. Login/Register With a Facebook Account
      * Description: A user would like to use their Facebook account to log into their InstructED account
      * Initial Conditions: The user must have a Facebook account, and must not have an InstructED account with the same email as their Facebook account
      * Input Data: Email/password for the Facebook account
      * Specifications: User can log into their account using Facebook Oauth
      * Procedure: Use one of our Facebook accounts to log in using Oauth
   6. Test Case 4: Testing Encryption of Passwords
      * Description: After adding a user to the database, the password should be encrypted to ensure security.
      * Initial Conditions: There must be a user in the database, and the person checking the database must know the unencrypted password for this user.
      * Input Data: A query to the database in order to see password info.
      * Specifications: The password a user submits is encrypted
      * Procedure: Add a user to the database, while knowing the password, then query the database to ensure that the password has been encrypted.
3. Overview (Sprint 2)
   1. Test Objectives
      * Verify that students can view upcoming assignments and their due dates
      * Verify students can view announcements for the classes that they are in
      * Verify that the calendar view gives access to the timetable for instructors and students
      * Verify that instructors can view the details of the courses taught by them
      * Verify that Instructors can make announcements for their courses
      * Verify that Admins can view details about all courses and their instructors
      * Verify that Admins will be able to approve and manage uploaded content
   2. Test Environment
      * 1: Local testing on a Linux system (Arch Linux), using the Brave browser (Chromium) and a locally hosted Postgresql database
      * 2: Non-Local testing using a Heroku based web server, and Postgresql database server
   3. Test Personnel
      * Kevin Cao
      * Andrew Jedlicka
   4. Acceptance Criteria
      * Every user should be able to view all information pertinent to them
   5. Noted Omissions
4. Test Cases (Sprint 2)
   1. Test Case 1: Students can view upcoming assignments
      * Description: A student who is enrolled in a class would like to view upcoming assignments
      * Initial Conditions: The account must be enrolled in a class, and at least one class that the student is in needs an assignment.
      * Input Data: None from the user except clicking on the correct widget. UserId to the database.
      * Specifications: The student can view upcoming assignments and the assignments are correctly assigned to a course they are in.
      * Procedure: Sign in using a student account which is enrolled in a course that has an assignment. Then click on the Upcoming Assignments widget and verify the output is correct.
   2. Test Case 2: Students can view announcements for their classes
      * Description: A student who is enrolled in a class would like to view class announcements
      * Initial Conditions: The account must be enrolled in a class, and the class must have at least one announcement made
      * Input Data: None from the user except clicking on the correct widget. UserId to the database.
      * Specifications: The student can view announcements for their courses
      * Procedure: Sign in using a student account which is enrolled in a course that has an announcement. Then click on the Announcements widget and verify the output is correct.
   3. Test Case 3: Calendar view gives access to the timeline for Students and Instructors
      * Description: A Student/Instructor would like to know the assignments for a certain day for all of their courses.
      * Initial Conditions: The account must be enrolled in or instructing a class.
      * Input Data: The user clicks on a day in the calendar. This and the UserId is sent to the database.
      * Specifications: The student/instructor can view assignments for a certain day
      * Procedure: Sign in using a student/instructor account that is enrolled in or instructing a class, then click on a day that is known to have an assignment and verify the output is correct.
   4. Test Case 4: Instructors can view details of their courses
      * Description: An instructor would like to view the details of a course they instruct.
      * Initial Conditions: The account must be an instructor, and be instructing a course.
      * Input Data: The courseID to the database, and the UserId to the database.
      * Specifications: The instructor can view the details of their course.
      * Procedure: Sign in using an instructor account that is instructing a class. Click on a class in the sidebar, then verify the details of the course are correct.
   5. Test Case 5: Verify Instructors can make announcements for their courses
      * Description: An instructor would like to make an announcement for a course they are instructing.
      * Initial Conditions: The account must be an instructor, and must be instructing a course.
      * Input Data: The courseId, and UserId to the database. As well as the announcement title and announcement text from the user.
      * Specifications: The instructor can make an announcement for a class they teach.
      * Procedure: Sign in using an instructor account that is instructing a class. Click on the make announcement widget and fill out all of the fields to make an announcement. Then sign in as a student for the class and verify the announcement is correct and is displayed.
   6. Test Case 6: Verify that Admins can view details about all courses and their instructors
      * Description: An admin would like to view the details of all courses and instructors.
      * Initial Conditions: The account must be an admin, and there must be at least one course in the database.
      * Input Data: None.
      * Specifications: The admin can view all courses and instructors for each course.
      * Procedure: Sign in using an admin account, then view the course list and verify the output is correct and matches the database.
5. Overview (Sprint 3)
   1. Test Objectives
      * Verify that instructors can create assignments
      * Verify that instructors can upload files to their respective courses
      * Verify that students can submit assignments in the format of a text box entry, files and media.
      * Verify that instructors can download submissions and grade students.
      * Verify that admins can view the uploaded files and approve/disapprove them
   2. Test Environment
      * 1: Local testing on a Linux system (Arch Linux), using the Brave browser (Chromium) and a locally hosted Postgresql database
      * 2: Non-Local testing using a Heroku based web server, and Postgresql database server, and Amazon S3 to store uploaded files
   3. Test Personnel
      * Kevin Cao
      * Andrew Jedlicka
   4. Acceptance Criteria
      * File upload and download should be operational for courses and assignments, as well as the ability to view the information about submissions and grades
   5. Noted Omissions
6. Test Cases (Sprint 3)
   1. Test Case 1: Verify Instructors can create assignments
      * Description: An instructor would like to create an assignment for their course
      * Initial Conditions: The account must be an instructor, and must be instructing a course.
      * Input Data: The assignment name, course, due date and time.
      * Specifications: Instructors can create assignments.
      * Procedure: Sign in using an instructor account, open the create assignment widget, fill all of the fields, then submit.
   2. Test Case 2: Verify Instructors can upload files to their courses
      * Description: An instructor would like to upload a file to a course they instruct.
      * Initial Conditions: The account must be an instructor and must be instructing a course.
      * Input Data: A file from the instructor’s machine.
      * Specifications: Instructors can upload files for their courses.
      * Procedure: Sign in using an instructor account, navigate to a class the instructor is teaching, go to the files or lecture videos tab, then click the upload button, choose a file, then upload it.
   3. Test Case 3: Verify that students can submit files, media, and text box entries for assignments
      * Description: A student would like to make a submission for an assignment.
      * Initial Conditions: The user must be a student and must be enrolled in a class.
      * Input Data: A type of submission such as a file upload, and a text entry.
      * Specifications: Students can submit files, media, and text box entries for assignments.
      * Procedure: Log in to a student account, navigate to an enrolled course with an assignment posted. Then click on the assignment, click on submit, then submit with the preferred submission method.
   4. Test Case 4: Verify that instructors can download submissions and grade students
      * Description: An instructor would like to download a submission then grade the given student based on that submission.
      * Initial Conditions: Must be signed into an instructor account that is teaching a course. That course must have at least one student and one assignment, and a submission to be evaluated.
      * Input Data: Grade Data.
      * Specifications: Instructors can download submissions and grade students.
      * Procedure: Sign in using an instructor account, navigate to an assignment and click on a student to see their submissions, then click on the submission to download it. Then enter a grade.
   5. Test Case 5: Verify that admins can view uploaded files and approve/disapprove them
      * Description: An admin would like to view all of the unapproved files and approve/disapprove of them.
      * Initial Conditions: User must be an admin, and there must be at least one unapproved file.
      * Input Data: Approval/Disapproval button press, as well as the button press for selecting the file and downloading it.
      * Specifications: Admins can view uploaded files and approve/disapprove of them.
      * Procedure: Sign in with an admin account, then open the UnapprovedFiles widget, click the view button next to the desired files to download and view it. Then click on the file in the list, which will bring up an approve/disapprove menu, click the preferred option.
7. Overview (Sprint 4)
   1. Test Objectives
      * Verify that instructors can search for assignments, announcements, and files
      * Verify that admins can search for students, instructors, courses, assignments, announcements, and files.
      * Verify that students can search for assignments, announcements, and files
   2. Test Environment
      * 1: Local testing on a Linux system (Arch Linux), using the Brave browser (Chromium) and a locally hosted Postgresql database.
      * 2: Non-Local testing using a Heroku based web server, and Postgresql database server, and Amazon S3 to store uploaded files
      * 3: Local testing on a Windows 10 system using Firefox and a locally hosted Postgresql database.
      * 4: Local testing on a MacOS system, using Safari, and a locally hosted Postgresql database.
   3. Test Personnel
      * Kevin Cao
      * Andrew Jedlicka
      * Shaun Trimm
   4. Acceptance Criteria
      * Every user should be able to search for files and users accessible to them
   5. Noted Omissions
8. Test Cases (Sprint 4)
   1. Test Case 1: Verify that instructors can search for assignments, announcements, and files
      * Description: An instructor would like to search for assignments, announcements, and files that they have access to
      * Initial Conditions: Must be an instructor account
      * Input Data: The name of the assignment, announcement, or file
      * Specification: Instructors can search for assignments, announcements, and files
      * Procedure: Sign in with the instructor account, open the search widget, and type in the name of the item you would like to search for.
   2. Test Case 2: Verify that admins can search for students, instructors, courses, assignments, announcements, and files.
      * Description: An admin would like to search for users, courses, assignments, announcements, and files.
      * Initial Conditions: Must be an admin account
      * Input Data: Then name of the user, course, assignment, announcement, or file
      * Specification: Admins can search for students, instructors, courses, assignments, announcements, and files
      * Procedure: Sign in with the admin account, open the search widget, and type in the name of the item you would like to search for.
   3. Test Case 3: Verify that students can search for assignments, announcements, and files
      * Description: A student would like to search for assignments, announcements, and files that they have access to.
      * Initial Conditions: Must be a student account
      * Input Data: The name of the assignment, announcement, or file
      * Specifications: Students can search for assignments, announcements, and files
      * Procedure: Sign in with the student account, open the search widget, and type in the name of the item you would like to search for.
9. Overview (Sprint 5)
   1. Test Objectives
      * Verify that users can chat with each other, including students with other students and instructors.
      * Verify that users can see message status (Delivered, Read, Typing) and the Online/Offline status of other users
   2. Test Environment
      * 1: Local testing on a Linux system (Arch Linux), using the Brave browser (Chromium) and a locally hosted Postgresql database.
      * 2: Non-Local testing using a Heroku based web server, and Postgresql database server, and Amazon S3 to store uploaded files
      * 3: Local testing on a Windows 10 system using Firefox and a locally hosted Postgresql database.
      * 4: Local testing on a MacOS system, using Safari, and a locally hosted Postgresql database.
   3. Test Personnel
      * Kevin Cao
      * Andrew Jedlicka
      * Shaun Trimm
   4. Acceptance Criteria
      * Chat is functional and allows users to see the message and user status
10. Test Cases (Sprint 5)
    1. Test Case 1: Verify that users can chat with each other, including students with other students and instructors.
       * Description: 2 users would like to chat with each other, one a student, and one an instructor.
       * Initial Conditions: Must have 2 accounts, one that is an instructor and one that is a student
       * Input Data: Chat input, and the selection of the other account.
       * Specifications: 2 users can chat with each other, one a student, and one an instructor.
       * Procedure: Sign into 2 separate sessions, one a student, one an instructor. Then message one account from another and repeat the messaging to verify chat functionality.
    2. Test Case 2: Verify that users can see message status (Delivered, Read, Typing) and the Online/Offline status of other users.
       * Description: Verify that users can see message status (Delivered, Read, Typing) and the Online/Offline status of other users
       * Initial Conditions: Must have 2 accounts that are able to message each other.
       * Input Data: Selecting the other user, and chatting enough to verify all functionality is working.
       * Specifications: Users can see message and user status.
       * Procedure: Sign into an account that has a user to chat with, verify the other account shows Online/Offline status in the chat, then send a message to verify the message status. Next receive a message from another account, read it, and check the other account to verify the status of that message.