PM4 - Black Box Test Plan
Segfaulters
Michael Kopczynski, Anushka Trivedi, Isha Devgan, Stuti Shah

NOTE: We have not implemented the project, so we have left the actual results column blank.

Test ID	Description	Expected Results	Actual Results
TestName, Test Author, Test Type	Preconditions, Steps, Test Inputs	Test Outputs	Actual Outputs
TestName: Focus Session Timer Start/Stop/End **TestAuthor:** Anushka **TestType:** Functional	Preconditions: PomBot downloaded, user has an account Steps: User goes to start a session. User then stops the session after desired time. If the user doesn't interfere in the stop, the session naturally ends. Test Inputs: Session that has already started, Values that determine whether the session has been started, and then stopped (essentially like a tracker).	The session should give a confirmation of the focus session being started. Additionally, once manually stopped, PomBot should also give a confirmation of that as well, and keep a log of the time between the focus sessions. If the focus session timer naturally exhausts, the data for this should be logged onto the user account as well.	
TestName: Edit Timer Duration **TestAuthor:** Anushka **TestType:** Functional	Preconditions: PomBot downloaded, user has an account, user has a focus session pre-set with a certain time amount. Steps: User goes to start a session, User clicks edit on the "Focus" aspect, User changes the desired minute setup time. Test Inputs: A value to change the duration of a focus session, A value to change the duration of a break session	After changing the desired time for a focus session, PomBot should confirm and save this time. The next time the user wants to start a focus session, this minute amount should be what is saved. The same applies for break duration- after the user changes the desired break minute time, PomBot should conserve and save this time for the next session.	
TestName: Add task to focus session **TestAuthor:** Anushka **TestType:** Functional	Preconditions: PomBot downloaded, user has an account, user is about to start a focus session, or has already completed one Steps: User goes to start a session, User is prompted to add a corresponding task to focus session, User adds the task in and saves it. Subtest: User has completed a focus session, User forgot to enter a corresponding task, PomBot prompts user to add a corresponding task, user adds it, and saves it.	This test has three different, but similar scenarios. The first one is that the user enters the name of a task before beginning a focus session. PomBot should confirm that the session was named. The second one is that the user skipped adding a task to a focus session, so after it ends, PomBot prompts them to write something, they write something, and PomBot confirms and saves the task. The last scenario is if the focus session has ended and has a task but it needs to be edited. This is a possible case if someone, for example, leaves out a detail for the task. In this case, the user can go to their profile, and click the corresponding focus session, and edit it to fix the focus session	

	Subtest: User has added a corresponding task to the focus session, but forgot to add something. User goes to the particular focus session, Goes to edit the corresponding task, and then saves it. Test Input: The name of a task to save to a focus session that has yet to be started, The name to add to a task that has already been completed, A name to change a focus session that has an already existing task	task.	
TestName: Timer Icon on Screen **TestAuthor:** Anushka **TestType:** UI Testing	Preconditions: User has downloaded PomBot, has an account for it, and has the extension screen/popup currently opened. Steps: User wants to start a work session, so the user opens the PomBot extension. User wants to click on the timer to start a focus session, so the user locates it on the screen. As testers, we must ensure that the icon is located on the screen to allow for proper usage of PomBot. Test Input: A screen for PomBot, and a value that represents the timer icon, and whether it is properly rendered on the screen.	After a user opens the PomBot extension, the screen must show a timer that allows them to start a work session. The purpose of this test is to ensure that the Timer Icon properly appears on screen, hence, we must use a value to represent this, and properly return whether the icon appears on screen or not.	
TestName: Timer counts down time properly **TestAuthor:** Anushka **TestType:** UI Testing	Preconditions: User has downloaded PomBot and has an account for it, User has the extension screen/popup opened, User has an ongoing focus/break session Steps: User has an ongoing session and wants to see how much time they have left. The user navigates to the timer displaying the current session timer, and they see time timer counting down in seconds. Test Input: A screen for PomBot with an ongoing session, and a value that represents the time remaining. The test ensures that the value is changing on the screen to represent time decreasing, and whether the changes are updating to the UI.	After a user opens the PomBot extension and has a current timer running. The test has a value representing the time (for minutes and seconds), and determines if the value is decreasing. The test also ensures that these values are properly rendering to the screen so that the user can view them.	
TestName:	Preconditions: PomBot downloaded,	When the user triggers actions like starting a	

Notifications Test **TestAuthor:** Stuti **TestType:** UI Testing	user has an account, and allow notifications is turned on Steps: Open the PomBot application, ensure that notifications are enabled in the app settings, perform actions that trigger notifications (e.g., start a timer, complete a task), verify that the expected notifications are received, check that the notification content is correct and formatted properly, confirm that the notifications appear at the appropriate times Test Input: perform actions that trigger notifications (e.g., start a timer, complete a task), verify that the expected notifications are received, check that the notification content is correct and formatted properly, confirm that the notifications appear at the appropriate times	timer or finishing a task, the app should send notifications at the right times. The notifications should be clear, accurate, and show the correct message, such as "Time's up!" or "Task completed." They should appear without delay and allow the user to interact with them.	
TestName: Change Timer Test **TestAuthor:** Stuti **TestType:** UI Testing	Preconditions: PomBot downloaded, user has an account, and they wish to change the time on the timer Steps: Open the PomBot application, start a timer or ensure a timer is active, navigate to the timer settings or time input, change the time (e.g., decrease or increase the timer value), save or apply the changes to the timer. Test Input: User changes the timer to a new value (e.g., 25 minutes to 15 minutes)	When the user changes the timer time, the new time should appear right away on the screen. The countdown should start from the new time without any problems. If the timer is paused and the time is changed, it should continue counting down from the updated time when resumed. The changes should happen instantly without errors or the need to restart the app.	
TestName: TestTeamSession Start **TestAuthor:** Isha **TestType:** Integration Testing	Preconditions: PomBot downloaded, user has an account, other users who want to join the session have an account Steps: User goes to start a session, User enters the emails of the users who will join the focus session. The users who are invited will check email associated with account if they received an invitation and can click the link to join the session Test Inputs: Emails/account names of all testers who will be in the same focus group, focus group/session name, time for each focus and break session set to 25 minutes and 5 minutes, respectively	The user who sends out the invite to the other users via email receives a ‘successfully sent’ message indicating that the invitations were sent on their end. The receivers of the invitation will receive either a notification or an email with a link to join the focus session. The link should redirect the user to the focus session, with their icon/avatar visible for everyone in the session to notify them that they have successfully joined. The focus group attendance should include every user/tester who was invited to the group and all those who accepted and joined versus those who were invited but did not join/declined.	
TestName: TestTeamSession End	Preconditions: PomBot downloaded, user has an account, other users who want to join the session have an account.	The user who finished the focus session receives a ‘successfully ended session message indicating that the session was	

TestAuthor: Isha **TestType:** Integration Testing	Team focus session was already initiated. Steps: User who initiated a session clicks the end session button. Test Inputs: Emails/account names of all testers who will be in the same focus group, focus group/session name, time for each focus and break session set to 25 minutes and 5 minutes, respectively	ended on their end. Those that were in the session but did not end it, will be navigated back to their settings page on the PomBot. When the session is ended, all users involved will receive a summary report of the focus and break sessions completed, and what tasks were tackled during the session. This report will also show the users that joined the session and for how long they individually joined the session.	
TestName: Group Creation **TestAuthor:** Anushka **TestType:** Integration Testing	Precondition: All aspiring members have PomBot downloaded and have accounts Steps: A ”leader” wants to make a group with their team to do regular focus/break sessions. The leader ensures all of the members have PomBot accounts, and can search through their names in the database. Once finding all of the members, the leader can invite them to a group, and from there forward, the group exists until it is deleted. Members should also be allowed to be added/removed. Test Inputs: A user for the account, other accounts that are to be added, accounts to be added after initial creation of the group, and predetermined accounts to test removing them from the group.	A team member wants to make a group to do regular focus sessions. The team member adds all of the users and invites them to be in the group. The app should allow them to add members in the group. After creation, users should still be allowed to be added to the group, as well as removed. There should be a value, which after performing add/removes, represents what the group should look like, and this should be compared with the actual group. Both of these values should match up to ensure the test passes.	