ID: BBS001
Name: Registered User Logs In
Description: A user logs in to the system with registered credentials.

Functional Requirements:
1 – System shall display a landing page to anonymous users.
3 – System shall require user to sign in to his account to use Bot!Battle!, and will verify the account's credentials against those on the database.
5 – System shall display the user homepage once user signs in.

Preconditions:
1. The user has a new browser window open.

Actors:
1. User has navigated to Bot!Battle! Landing Page.

Main Course:
1. System displays landing page to user according to landing page wireframe.
2. User enters username in the username text box.
3. User enters password in the password text box.
4. User clicks the Login button.
5. System requests authorization token from the database.
6. Database returns authorization token to User.
7. User is redirected to User Homepage.

Preconditions:
1. User has signed in to their account.
2. User has navigated to User Homepage.

Alternative Courses:
5.1 – User clicks Login without supplying both pieces of credentials.
1. Landing Page displays a message stating “Both username and password are required”.
2. Resume at Step 2.
6.1 – Bot!Battle! Landing Page fails to connect to database.
1. Landing Page displays a message stating “Unable to connect to database”
2. Resume at Step 2.
6.2 – User enters username and/or password incorrectly:
1. Landing Page displays an error message stating “Incorrect username or password”.
2. Resume at Step 2.

ID: BBS002
Name: Contests List Page
Description: A user selects a contest from the Contests List Page.
Functional Requirements:
6 – The user homepage shall display the following: link to the lessons page, link to the challenges page, link to the contests page, link to the Contact Us page, link to the About Bot!Battle! page, link to the My Account page, and a link to the Administrator Page if the user is an administrator.
27 – System shall maintain in the database for each contest a contest date, a challenge, contest results, contest status, and uploaded bots.
28 – Contests list page shall list contests with a contest status of "upcoming" on the contests list page in chronological order according to their contest date.
29 – Contests list page shall list contests with a contest status of "completed" or "in progress" in reverse chronological order according to their contest date.
30 – Contests list page shall allow user to select a contest, redirecting the user to the selected contest page.

Actors:
1. USER
Preconditions:
1. User is logged in.
Main Flow:
1. System displays user homepage according to user homepage wireframe.
2. USER clicks Contests.
3. System redirects user to Contest List Page.
4. Contests list page retrieves a list of contests from the database.
5. Contests list page sorts list of upcoming contests in chronological order.
6. Contests list page sorts list of in progress and completed contests in reverse chronological order.
7. Contests list page displays upcoming contests according to contests_list_page wireframe.
8. Contests list page displays in progress and completed contests according to contests_list_page wireframe.
9. User selects a link to a contest page.
10. User is redirected to the selected contest page.
Post Condition:
User has navigated to a Contest Page.
Alternate Flows:
4.1 – Contests list page fails to connect to the database.
1. Contests list page displays a message in the center of the page which states "Error: Unable to connect to the Bot!Battle! database".
7.1 – There are no upcoming contests retrieved.
1. Contests list page displays message where upcoming contests would be displayed which states "There are no upcoming contests at this time".
2. Resume at Step 4.
8.1 – There are no in progress or completed contests.
1. Contests list page displays message where completed contests would be displayed "There are no in progress or completed contests at this time".
2. Resume at Step 6.

ID: BBS003
Name: User Views Contest
Description: Any user may view contests.

Functional Requirements:
31 - System shall display the selected contest's challenge, name, contest date, and contest status on the selected contest page.

Actors:
USER

Preconditions: User has navigated to the Contest Page.

Main Flow:
1. CONTEST_PAGE requests the contest's challenge, name, date, and status from DATABASE.
2. DATABASE returns contest's challenge, name, date, and status to CONTEST_PAGE.
3. CONTEST_PAGE displays fields according to contest page wireframe to USER.

Post Conditions:
Web system displays dynamically generated Contest Page.

Alternate Flows:
1.1 – CONTEST_PAGE fails to connect to the database.
1. CONTEST_PAGE displays an error dialogue which states "Error: Unable to connect to the Bot!Battle! Database".

ID: BBS003.5
Name: User Views Contest
Description: Any user may participate in contests.

Functional Requirements:
32 - Contest page shall allow user to upload a bot for the selected contest if the contest status is “upcoming”.

Actors:
USER

Preconditions: User has navigated to the Contest Page.

Main Flow:
1. USER selects the link to upload bot.
2. CONTEST_PAGE requests bot names and id's for user id and contest challenge id from DATABASE.
3. DATABASE returns bot names and id's to CONTEST_PAGE.
4. CONTEST_PAGE displays Select Bot dialog with bots from DATABASE.
5. User selects bot.
6. User clicks Select Bot.
7. CONTEST_PAGE sends selected bot's id and contest id to DATABASE.
8. DATABASE creates entry in contest for user's id and selected bot's id.

Post Conditions:
If necessary, the user's bot is uploaded to the database.
The user and user's bot are entered into the contest.

Alternate Flows:
1.1 – CONTEST_PAGE fails to connect to the database.
1. CONTEST_PAGE displays an error dialogue which states “Error: Unable to connect to the Bot!Battle! Database”.
1.2 – Contest status is 'in progress' or 'complete'.
1. CONTEST_PAGE displays a message which states “you may not upload bots for contests that are in progress or complete!”.
1.3 – User selects the link to view contest's challenge.
1. USER is redirected to the challenge page.
4.1 – The user has not uploaded any bots for the contest's challenge.
1. Select Bot button is grayed out.
2. Resume at Step 9.
6.1 – User selects Upload New Bot.
1. System redirects user to Upload Bot Page (BBS023).
2. CONTEST_PAGE sends newbot's id and contest id to DATABASE.
3. Resume at Step 11.
8.1 – User has previously submitted a bot for the contest.
1. DATABASE replaces entry in contest for user's id with selected bot's id.

ID: BBS004
Name: View Contest Results
Description: USERs may view games from contest results.

Functional Requirements:
33 – CONTEST_PAGE shall allow USERs to view contest results once the contest status is “in progress” or “completed.”

Actors:
USER

Preconditions:
USER has navigated to the Contest Page.
Contest status is either In Progress or Complete.

Main Flow:
1. CONTEST_PAGE requests the list of contest results from the DATABASE.
2. DATABASE returns list of contest results to CONTEST_PAGE.
3. CONTEST_PAGE displays the list of contest results as specified in the wireframe to USER.
4. USER selects a Contest Result from CONTEST_PAGE.
5. USER is redirected to GAMEPAGE.

Post Conditions:
USER has navigated to a GAMEPAGE

Alternate Flows:
1.1 – CONTEST_PAGE fails to connect to the DATABASE.
1. CONTEST_PAGE displays a message which states “Error: Unable to connect to the Bot!Battle! Database” on ERRORPAGE.
3.1 – There are no contest results to display.
1. CONTEST_PAGE displays a message which states “Error: No matches have been run yet” on ERRORPAGE.

ID: BBS005
Name: Run Contest in Series
Description: An administrator may run contests.

Functional Requirements:
34 - Contest page shall allow an administrator to run a contest.
35 - System shall allow contest results to be broadcast.

Actors:
ADMIN

Preconditions:
User has navigated to the Contest Page.
Contest status is 'In Progress'.

Main Flow:
1. CONTEST_PAGE displays link to run contest according to contest page wireframe to ADMIN.
2. ADMIN selects link to run contest.
3. CONTEST_PAGE displays contest run dialogue box according to run contest wireframe.
4. ADMIN selects “Individual Games in Series.”
5. ADMIN clicks “Run Tournament.”
6. System adds Contest ID to enqueued contests table in DATABASE.
7. Game Server runs the contest's challenges.
8. Game Server adds contest challenges to MATCH_TURNS and CONTEST_MAP.
9. Game Server sets the finished field of the contest record to 1.

Post Conditions:
The results of the challenges are added to MATCH_TURNS and CONTEST_MAP.

Alternate Flows:
6.1 – ADMIN selects “Run Contest” without selecting how to run the contest.
1. Contest run dialogue box displays a message which states “You must select a contest run option!”.
2. Resume at Step 5.
6.2 – ADMIN selects “Cancel.”
1. Contest run dialogue box is closed.
2. Resume at Step 2.
7.1 – Contest run dialogue box fails to connect to the game server
1. Contest run dialogue box displays a message which states “Error: Unable to connect to game server!”.

ID: BBS006
Name: Admin Navigates to Edits Contest
Description: Administrators may edit contests.

Functional Requirements:
27 – System shall maintain in the database for each contest a name, a contest date, a challenge, contest results, contest status, and uploaded bots.
31 – System shall display the selected contest's challenge, name, contest date, and contest status on the selected contest page.
36 – If user is an administrator, the contest page shall display a link for "Add New Contest", a link for "Edit Contest", and a link for "Delete Contest".

Actors:
ADMIN

Preconditions:
ADMIN navigates to the Contest Page.

Main Flow:
1. CONTEST_PAGE requests the contest's challenge, name, date, and status from DATABASE.
2. DATABASE returns contest's challenge, name, date, and status to CONTEST_PAGE.
3. CONTEST_PAGE displays fields according to contest page wireframe to ADMIN.
4. CONTEST_PAGE displays administrator options according to contest page wireframe to ADMIN.
5. ADMIN selects "Edit Contest."
6. ADMIN is redirected to Edit Contest Page.

Post Conditions: System displays the Edit Contest Page.

Alternate Flows:
1.1 – CONTEST_PAGE fails to connect to the database.
1. CONTEST_PAGE displays a message which states "Error: Unable to connect to the Bot!Battle! Database".
5.1 – User selects link to 'Delete Contest'.
1. CONTEST_PAGE displays delete page dialogue box according to delete contest wireframe.
2. ADMIN selects 'Yes'.
3. Delete page dialogue box submits delete contest request to DATABASE.
4. Delete page dialogue box is closed.
5. ADMIN is redirected to CONTESTS_LIST_PAGE.
5.1.3 – ADMIN selects 'No'.
1. Delete page dialogue box is closed.
2. Resume at Main Flow step 4.
5.2 – ADMIN selects 'Add New Contest'.
1. ADMIN is redirected to New Contest Page.

ID: BBS007
Name: User Edits Account Information.
Description: The user chooses to view or edit details of their account.

Functional Requirements:
6 – The user homepage shall display the following: link to the lessons page, link to the challenges page, link to the contests page, link to the Contact Us page, link to the About Bot!Battle! page, link to the My Account page, and a link to the Administrator Page if the user is an administrator.

Actors:
USER

Preconditions:
USER has signed in to their account.

Main Flow:

1. System displays user homepage according to user homepage wireframe.
2. USER clicks on My Account.
3. System redirects user to My Account page.
4. My Account Page requests the user account's name, user name, and email from the database.
5. DATABASE returns user account's name, user name, and email to My Account page.
6. My Account Page displays fields according to My Account wireframe.
7. My Account Page fields are populated with corresponding data from DATABASE.
8. User enters account information updates into appropriate textboxes.
9. User enters current password in Confirm Current Password textbox.
10. User clicks Update Account Information button.
11. My Account Page sends updated details to Database.
12. My Account Page displays confirmation dialogue to User.

Preconditions:
USER has navigated to My Account page.

Alternate Flows:
4.1 – My Account Page fails to connect to the Database.
1. My Account Page displays a message on Dialog box which states “Error: Unable to connect to the Bot!Battle! Database”.
10.1 – No fields have been changed.
1. My Account Page displays a message on Dialog box which states “No fields have been changed”.
2. Resume at Step 8.
11.1 – User has not entered the correct password in Confirm Current Password textbox.
1. My Account Page displays a message on Dialog box which states “Password in Confirm Current Password is not correct”.
2. Resume at Step 9.
11.2 – User has entered a new password which does not match the password in Confirm New Password.
1. My Account Page displays a message on Dialog box which states “New Password and Confirm New Password do not match”.
2. Resume at Step 8.

ID: BBS008
Name: Registration
Description: Anonymous User registers with the system.

Functional Requirements:
2 – System shall allow user to register an account to use Bot!Battle!, and will store their specified account credentials on the database.

Actors: ANONYMOUS USER

Preconditions:
ANONYMOUS USER is at landing page.

Main Flow:
1. ANONYMOUS USER clicks “Register” on landing page.
2. System displays registration_page according to registration page wireframe.
3. ANONYMOUS USER enters the required information.
4. ANONYMOUS USER clicks Register Account button.
5. Registration page verifies all input has appropriate information.
6. Registration page stores information on the database.

Post Conditions:
1. ANONYMOUS USER has an account to use the system.
2. ANONYMOUS USER is now a USER.

Alternate Flows:
5.1 – The username, password, or email contains illegal characters that could be malicious.
1. Registration page displays message “Error: you are not allowed to characters such as '<,>* ' as part of a username, password, or email”.
2. Resume at 3.
5.2 – An input field is left blank.
1. Registration page displays message “All fields must be filled out”
2. Resume at Step 3.
5.3 – The password confirmation input does not match the password input.
1. Registration page displays message “Password confirmation does not match entered password”
2. Resume at Step 3.
6.1 – The database cannot be contacted.
1. Registration page displays message "Error: Database cannot be opened at this moment”.
6.2 – The email address already exists in the database.
1. Registration page displays message “There is already an account registered with that email address”.
2. Resume at Step 3.
6.3 – The requested username already exists in the database.
1. Registration page displays message “There is already an account registered with that username”.
2. Resume at Step 3.

ID: BBS009
Name: Select from Challenges List Page
Description: A user selects a challenge from the Challenges List Page.
Functional Requirements:
6 – The user homepage shall display the following: link to the lessons page, link to the challenges page, link to the contests page, link to the Contact Us page, link to the About Bot!Battle! page, link to the My Account page, and a link to the Administrator Page if the user is an administrator.
18 – System shall maintain in the database for each challenge a name, a category, a difficulty, completion information, published bots, and relevant lessons.
19 – System shall display all challenges on challenges list page.
20 – Challenges list page shall group challenges by category first, then sorted on difficulty.
21 – Challenges list page shall allow user to select a challenge, redirecting user to the selected challenge page.

Actors: User
Preconditions: User is logged in.
Main Flow:
1. System displays user homepage according to user homepage wireframe.
2. USER clicks Challenges.
3. System redirects user to Challenges List Page.
4. CHALLENGES_LIST_PAGE requests list of challenges from the DATABASE.
5. DATABASE returns list of challenges to CHALLENGES_LIST_PAGE.
6. CHALLENGES_LIST_PAGE sorts list of challenges by category.
7. CHALLENGES_LIST_PAGE sorts list challenges by difficulty.
8. CHALLENGES_LIST_PAGE displays upcoming contests according to contests_list_page wireframe.
9. User selects a link to a CHALLENGE_PAGE.
10. User is redirected to the selected CHALLENGE_PAGE.

Post Condition:
User has navigated to a Challenge Page.
Alternate Flows:
4.1 – CHALLENGES_LIST_PAGE fails to connect to the database.
1. CHALLENGES_LIST_PAGE displays a message “Error: Unable to connect to the Bot!Battle! database”.

5.1 – There are no challenges retrieved.
1. CHALLENGES_LIST_PAGE displays a message “There are no challenges at this time”.

ID: BBS010
Name: User Views Challenge Page
Description: User has navigated to and is viewing the challenge page. Users may complete challenges.

Functional Requirements:
22 - System shall display the selected challenge's name, category, relevant lessons and completion information on the selected challenge page.
24 – Challenge page shall display links to testing arenas, where user's bot for the selected challenge can be tested.

Actors:
USER

Preconditions: User has navigated to the Challenge Page.

Main Flow:
1. CHALLENGE_PAGE requests the challenge's name, category, difficulty, completion info, and relevant lessons from DATABASE.
2. DATABASE returns challenge's name, category, difficulty, completion info, and relevant lessons to CHALLENGE_PAGE.
3. CHALLENGE_PAGE displays fields according to challenge page wireframe.
4. USER selects the 'Upload Bot for Challenge' link.
5. System redirects user to Upload Bot Page (BBS023).

Post Conditions:
System displays dynamically generated Challenge Page

Alternate Flows:
1.1 – System failed to connect to the database.
1. System displays CHALLENGE_PAGE with a message in the center of page which states "Error: Unable to connect to the Bot!Battle! Database".
4.1 – USER selects link for a relevant lesson.
1. USER is redirected to the selected lesson.

ID: BBS011
Name: Admin Navigates to Edits Challenge
Description: Administrators may edit challenges.

Functional Requirements:
18 – System shall maintain in the database for each challenge a name, a category, a difficulty, completion information, published bots, and relevant lessons.
22 – System shall display the selected challenge's name, category, relevant lessons and completion information on the selected challenge page.
26 – If user is an administrator, the contest page shall display a link for "Add New Challenge", a link for "Edit Challenge", and a link for "Delete Challenge".

Actors: ADMIN

Preconditions:
ADMIN has navigated to the Challenge Page.

Main Flow:
1. CHALLENGE_PAGE requests the challenge's name, category, difficulty, completion information, published bots and relevant lessons from DATABASE.
2. CHALLENGE_PAGE displays fields according to challenge page wireframe to ADMIN.
3. CHALLENGE_PAGE displays administrator options according to challenge page wireframe to ADMIN.
4. ADMIN selects link to edit challenge.
5. CHALLENGE_PAGE redirects ADMIN to Edit Challenge Page for editing challenge.

Post Conditions: ADMIN is redirected to Edit Challenge Page.

Alternate Flows:
1.1 – CHALLENGE_PAGE fails to connect to the database.
1. CHALLENGE_PAGE displays a message on Dialog box which states
"Error: Unable to connect to the Bot!Battle! Database".

ID: BBS012
Name: Admin Edit Challenge
Description: Administrators may edit challenges

Functional Requirements:
26 – Challenge page shall allow administrators to add, edit, or delete challenge.

Actors:
USER

Preconditions:
User has navigated to the Edit Challenge Page.
User is an administrator.

Main Flow:
1. Edit Challenge Page requests challenge's name, category, difficulty, completion information and relevant lessons from DATABASE.
2. DATABASE returns challenge's name, category, difficulty, completion information and relevant lessons to Edit Challenge Page.
3. Edit Challenge Page displays fields to edit challenge according to challenge details wireframe.
4. Edit Challenge Page fields are populated with corresponding data from DATABASE.
5. User makes changes to fields in Edit Challenge Page.
6. User selects 'Save'.
7. Challenge Details page submits update contest request to DATABASE.
8. User is redirected to CHALLENGE_PAGE.

Post Conditions:
CHALLENGE_PAGE reflects submitted changes.

Alternate Flows:
1.1 – Edit Challenge Page fails to connect to the database.
1. Edit Challenge Page displays a message on Dialog box which states
"Error: Unable to connect to the Bot!Battle! Database".
6.1 – Edit Challenge Page has at least one blank field.
1. Edit Challenge Page displays a message stating "No fields may be left blank!".
2. Resume at Step 5.
6.2 – Edit Challenge Page has no changed fields.
1. User is redirected to CHALLENGE_PAGE.

ID: BBS013
Name: User selects Lesson
Description: A user selects a lesson from the Lessons List Page.
Functional Requirements:

6 – The user homepage shall display the following: link to the lessons page, link to the challenges page, link to the contests page, link to the Contact Us page, link to the About Bot!Battle! page, link to the My Account page, and a link to the Administrator Page if the user is an administrator.
12 – System shall maintain in the database for each lesson a name, a category, a difficulty, lesson text, and an optional video.
13 – System shall display all lessons on the lessons list page.
14 – Lessons list page shall group lessons by category first, then by difficulty.
15 – Lessons list page shall allow user to select a lesson, redirecting the user to the selected lesson page.

Actors: User
Preconditions: User is logged in.
Main Flow:
1. System displays user homepage according to user homepage wireframe.
2. USER clicks My Account.
3. System redirects user to My Account page.
4. LESSONS_LIST_PAGE requests list of lessons from the DATABASE.
5. DATABASE returns list of challenges to LESSONS_LIST_PAGE.
6. LESSONS_LIST_PAGE sorts list of lessons by difficulty.
7. LESSONS_LIST_PAGE sorts list of lessons by category.
8. LESSONS_LIST_PAGE displays lessons according to the lessons_list_page wireframe.
9. User selects a link to a LESSON_PAGE.
10. User is redirected to the selected LESSON_PAGE.

Post Condition:
User has navigated to a Lesson Page.
Alternate Flows:
4.1 – LESSONS_LIST_PAGE fails to connect to the database.
1. LESSONS_LIST_PAGE displays a message on dialog box, “Error: Unable to connect to the Bot!Battle! database”.
5.1 – There are no lessons retrieved.
1. LESSONS_LIST_PAGE displays a message in the on dialog box, “There are no lessons at this time”.

ID: BBS014
Name: User Views Lesson
Description: Users may view lesson pages.

Functional Requirements:
16 – System shall display the selected lesson's category, name, lesson text, and optional video on the selected lesson page.

Actors:
USER
CONTEST_PAGE
DATABASE
CENTER_OF_PAGE

Preconditions: User has navigated to the Lesson Page.

Main Flow:
1. LESSON_PAGE requests the lesson's, name, category, lesson text, and video from DATABASE.
2. Database responds with the requested information.
3. LESSON_PAGE displays fields according to lesson page wireframe.

Post Conditions:
User may read the lesson.

Alternate Flows:
1.1 – LESSON_PAGE fails to connect to the database.
1. LESSON_PAGE displays a message on dialog box, states
"Error: Unable to connect to the Bot!Battle! Database".
2.1 – DATABASE returns a null value for the video.
1. LESSON_PAGE does not display a video element.

ID: BBS015
Name: Admin Navigates to Edits Lesson
Description: Administrators may edit lessons.

Functional Requirements:
16 – System shall display the selected lesson's category, name, lesson text, and optional video on the selected lesson page.
17 – If user is an administrator, the lesson page shall display a link for "Add New Lesson", a link for "Edit Lesson", and a link for "Delete Lesson".

Actors:
ADMIN

Preconditions:
ADMIN has navigated to the Lesson Page.

Main Flow:
1. LESSON_PAGE requests the lesson's, name, category, lesson text, and video from DATABASE.
2. DATABASE returns lesson's name, category, lesson text, and video to LESSON_PAGE.
3. LESSON_PAGE displays fields according to lesson page wireframe.
4. LESSON_PAGE displays administrator options according to lesson page wireframe.
5. ADMIN selects link to edit lesson.
6. LESSON_PAGE redirects ADMIN to Lesson Details page for editing lesson.

Post Conditions:
LESSON_DETAIL_PAGE is displayed.

Alternate Flows:
1.1 – LESSON_PAGE fails to connect to the database.
1. LESSON_PAGE displays a message on dialog box, "Error: Unable to connect to the Bot!Battle! Database".

ID: BBS016
Name: Admin New Lesson Page
Description: Administrator adds lesson to Database.

Functional Requirements:
9 – Administrator page shall allow administrator to specify a new lesson's category, name, difficulty, lesson text, and optional video.

Actors:
ADMIN

Preconditions:
ADMIN has navigated to the New Lesson Page.

Main Flow:
1. New Lesson Page displays content according to lesson details wireframe.
2. New Lesson Page fields are populated with corresponding data from DATABASE.
3. ADMIN makes changes to fields in New Lesson Page.
4. ADMIN selects 'Save'.
5. New Lesson Page sets Admin Creator field to ADMIN's username.
6. New Lesson Page submits new lesson request to DATABASE.
7. ADMIN is redirected to LESSON_PAGE.

Post Conditions:
1. The database contains the created lesson.
2. ADMIN has navigated to the newly created LESSON_PAGE.

Alternate Flows:
4.1 – New Lesson Page has at least one blank field.
1. New Lesson Page displays a message on dialog box stating "No fields may be left blank!".
2. Resume at Step 5.
5.1 – New Lesson Page fails to connect to the database.
1. New Lesson Page displays a message on dialog box which states
"Error: Unable to connect to the Bot!Battle! Database".

ID: BBS017
Name: Admin Adds New Challenge
Description: Administrator adds task to database.

Functional Requirements:
10 – Administrator page shall allow administrator to specify a new challenge's category, name, difficulty, completion information and relevant lessons from the administrator page.

Actors:
ADMIN

Preconditions:
ADMIN has navigated to the New Challenge Page.

Main Flow:
1. New Challenge Page displays content according to challenge details wireframe.
2. New Challenge Page fields are populated with corresponding data from DATABASE.
3. ADMIN makes changes to fields in New Challenge Page.
4. ADMIN selects 'Save'.
5. New Challenge page sets Admin Creator field to ADMIN's username.
6. New Challenge Page submits new challenge request to DATABASE.
7. ADMIN is redirected to CHALLENGE_PAGE.

Post Conditions:
1. The database contains the created lesson.
2. ADMIN has navigated to the newly created CHALLENGE_PAGE.

Alternate Flows:
4.1 – New Challenge Page has at least one blank field.
1. New Challenge Page displays a message stating "No fields may be left blank!" on dialog box.
2. Resume at Step 5.
5.1 – New Challenge Page fails to connect to the database.
1. New Challenge Page displays a message on dialog box which states
"Error: Unable to connect to the Bot!Battle! Database".

ID: BBS0018
Name: Admin Adds New Contest
Description: Administrator adds contest to database.

Functional Requirements:
11 – Administrator page shall allow administrators to specify a new contest's contest date, name, and challenge.

Actors:
ADMIN

Preconditions:
ADMIN has navigated to the New Contest Page.

Main Flow:
1. New Contest Page displays content according to contest details wireframe.
2. New Contest Page fields are populated with corresponding data from DATABASE.
3. Admin makes changes to fields in New Contest Page.
4. Admin selects 'Save'.
5. New Contest page sets Admin Creator field to ADMIN's username.
6. New Contest Page submits new contest request to DATABASE.
7. ADMIN is redirected to CONTEST_PAGE.

Post Conditions:
1. The database contains the created lesson.
2. ADMIN has navigated to the newly created CONTEST_PAGE.

Alternate Flows:
4.1 – New Contest Page has at least one blank field.
1. New Contest Page displays a message stating “No fields may be left blank!” on dialog box.
2. Resume at Step 5.

5.1 – New Contest Page fails to connect to the database.
1. New Contest Page displays a message in the center of the page which states
“Error: Unable to connect to the Bot!Battle! Database”.

ID: BBS019
Name: User Signs Out
Description: The user signs out of Bot!Battle!.

Functional Requirements:
4 – System shall allow user to sign out of their account.

Actors:
USER

Preconditions:
USER's browser displays user_homepage.

Main Flow:

1. USER clicks "Log out" link.
2. System sets authorization token to null.
3. System redirects USER to landing_page.

Postconditions:
USER is logged out.

ID: BBS020
Name: Admin Edits Contest
Description: Administrators may edit contests.

Functional Requirements:
36 – Contest page shall allow administrators to add, edit, or delete contest.

Actors:
ADMIN

Preconditions:
ADMIN has navigated to Edit Contest Page.

Main Flow:
1. Edit Contest Page requests contest's challenge, name, date, and status from DATABASE.
2. DATABASE returns contest's challenge, name, date, and status to Edit Contest Page.
3. Edit Contest Page displays fields to edit contest according to contest details wireframe.
4. Edit Contest Page fields are populated with corresponding data from DATABASE.
5. ADMIN makes changes to fields in Edit Contest Page.
6. ADMIN clicks 'Save'.
7. Contest Details page submits update contest request to DATABASE.
8. ADMIN is redirected to CONTEST_PAGE.

Post Conditions:
CONTEST_PAGE reflects submitted changes.

Alternate Flows:
1.1 – Edit Contest Page fails to connect to the database.
1. Edit Contest Page displays a message on a dialog box, “Error: Unable to connect to the Bot!Battle! Database”.
6.1 – Edit Contest Page has at least one blank field.
1. Edit Contest Page displays a message on dialog box that states “No fields may be left blank!”.
2. Resume at Step 5.
6.2 – Edit Contest Page has no changed fields.
1. User is redirected to CONTEST_PAGE.

ID: BBS021
Name: Admin Edits Lesson
Description: Administrators may edit lessons.

Functional Requirements:
17 – If user is an administrator, the lesson page shall display a link for “Add New Lesson”, a link for “Edit Lesson”, and a link for “Delete Lesson”.

Actors:
ADMIN

Preconditions:
ADMIN has navigated to the Edit Lesson Page.

Main Flow:
1. Edit Lesson Page requests the lesson's, name, category, lesson text, and video from DATABASE.
2. DATABASE returns lesson's name, category, lesson text, and video to Edit Lesson Page.
3. Edit Lesson Page displays the Edit Lesson Page according to lesson details wireframe.
4. ADMIN changes fields on the Edit Lesson Page.
5. ADMIN clicks 'Save'.
6. Edit Lesson Page submits changes to DATABASE.
7. ADMIN redirected to lesson page.

Post Conditions:
Lesson page reflects submitted changes.

Alternate Flows:
1.1 – Edit Lesson Page fails to connect to the database.
1. Edit Lesson Page displays a message on dialog box which states
“Error: Unable to connect to the Bot! Battle! Database”.
6.1 – Edit Lesson Page has at least one blank field.
1. Edit Lesson Page displays a message on dialog box stating “No fields may be left blank!”
2. Resume at Step 5.
6.2 – Edit Lesson Page has no changed fields.
1. ADMIN is redirected to LESSON_PAGE.

ID: BBS022
Name: API Requests Game Information.
Description: An API call is received requesting map or match information.

Functional Requirements:
37 – System shall provide an API for uploading and accessing saved matches and maps.

Actors:
API USER

Preconditions:
API USER has requested map, initialization, or turn information from a serialized JSON string.

Main Flow:
1. System processes request from HTML POST message.
2. System requests the requested map, initialization, or turn information from the DATABASE.
3. DATABASE returns map, initialization, or turn information to System.
4. A JSON object is constructed from the returned map, initialization, or turn information.
5. The JSON object is serialized as a string.
6. The serialized string is returned to API USER.

Post Conditions:
API USER has a string they can deserialize as a JSON object

Alternate Flows:
3.1 – DATABASE cannot find requested map initialization, or turn information.
1. System returns the message to API USER which states “Error: Resource does not exist!” along with the request string.

ID: BBS023
Name: User Uploads Bot
Description: A user uploads a bot to the system for a challenge or contest.

Functional Requirements:
37 – System shall provide an API for uploading and accessing saved matches and maps.

Actors:
USER

Preconditions:
USER has chosen to upload a bot.

Main Flow:
1. System displays select file dialog to USER.
2. USER specifies the path to bot file on hard drive.

Post Conditions:
API USER has a string they can deserialize as a JSON object

Alternate Flows:
3.1 – DATABASE cannot find requested map initialization, or turn information.
1. System returns the message to API USER which states “Error: Resource does not exist!” along with the request string.