Bootstrap Buddies

DATS Chess

Chris Tatlonghari, Anthony DelCimmuto

Joseph Snider, Rouba Al-Khasawneh

Test Plan

I. Summary of Overall Test Plan

The three most important components of our web application were the focus of our testing approach, namely, the chess engine, user registration, and database. We also included full system testing to ensure that these major components worked well with each other. In order to test normal as well as abnormal test cases we will make use of Perft, DBUnit, and other software testing tools.

II. Test Case Descriptions

CE1.1 **Chess Engine Test 1**

CE1.2 The purpose of this test is to ensure the engine plays legal moves.

CE1.3 This test will ensure that all legal moves are possible and only legal moves are playable. This will allow for a complete game.

CE1.4 Input: FEN strings.

CE1.5 Output: String of the form “bestmove xxxxx” per UCI.

CE1.6 Normal

CE1.7 Blackbox

CE1.8 Functional

CE1.9 Unit

CE2.1 **Chess Engine Test 2**

CE2.2 The purpose of this test is to ensure that the engine gives an evaluation.

CE2.3 The engine must give a value for the current state of the game (evaluation) given a current position.

CE2.4 Input: FEN string.

CE2.5 Output: Float or string “Mx” where x is an integer.

CE2.6 Normal

CE2.7 Blackbox

CE2.8 Functional

CE2.9 Unit

CE3.1 **Chess Engine Test 3**

CE3.2 The purpose of this test is to examine the accuracy of the engine's evaluation.

CE3.3 The evaluation of the DATS Chess engine will be compared with that of Stockfish 14.1’s evaluation to compare accuracy.

CE3.4 Input: 2 Floats

CE3.5 Output: A Percentage

CE3.6 Normal & Abnormal

CE3.7 Whitebox

CE3.8 Performance

CE3.9 Unit

CE4.1 **Chess Engine Test 4**

CE4.2 The purpose of this test is to ensure that the engine has a reasonable speed.

CE4.3 The engine should return the best move in less than 4 seconds upon request.

CE4.4 Input: A String for move request

CE4.5 Output: Bool, True for returning in less than 4 seconds.

CE4.6 Normal

CE4.7 Whitebox

CE4.8 Performance

CE4.9 Unit

R1.1 **Registration Test 1**

R1.2 This test will ensure the input of our sign in registration page.

R1.3 This test will validate that a user can sign in if they enter a valid username and password.

R1.4 Input: 2 strings, a username and a password

R1.5 output: successful sign in, and navigation

R1.6 normal

R1.7 blackbox

R1.8 functional

R1.9 Unit testing

R2.1 **Registration Test 2**

R2.2 This test will ensure the input of our sign up registration page.

R2.3 This test will validate that a user can sign up if they enter valid data.

R2.4 Input: 3 strings, a username, a password, and a confirm password

R2.5 Output: successful sign up, and navigation

R2.6 normal

R2.7 blackbox

R2.8 functional

R2.9 unit testing

R3.1 **Registration Test 3**

R3.2 This test will ensure that a user cannot sign in when inputting incorrect data (username / password).

R3.3 This test will validate that if a user inputs the incorrect data, then do not sign them in and display proper warning indications.

R3.4 Input: 2 strings, a username and a password

R3.5 output: unsuccessful sign in, and warning indications

R3.6 normal

R3.7 blackbox

R3.8 functional

R3.9 Unit testing

R4.1 **Registration Test 4**

R4.2 This test will ensure the functionality of sign up errors.

R4.3 This test will validate that a user is not able to sign up if they enter invalid username or password (i.e. already used username or not a strong enough password) and display proper warning indications.

R4.4 Input: 3 strings, a username, a password, and a confirm password

R4.5 Output: unsuccessful sign in, and warning indications

R4.6 normal

R4.7 blackbox

R4.8 functional

R4.9 unit testing

DB1.1 **Database Test 1**

DB1.2 The purpose of this test is to show that a DB user can retrieve games.

DB1.3 The database should correctly return all the games by a given user according to the following parameters: Date Played, Piece Color, and OpponentID.

DB1.4 Input: SQL query with parameters DateTime, Bool, and integer

DB1.5 Output: A list of games

DB1.6 Normal

DB1.7 Blackbox

DB1.8 Functional

DB1.9 Unit

DB2.1 **Database Test 2**

DB2.2 The purpose of this test is to show that a DB user can retrieve user info.

DB2.3 The database should correctly return the user info given a certain user ID.

DB2.4 Input: SQL Query with an integer parameter

DB2.5 Output: User’s Name, email, and password

DB2.6 Normal

DB2.7 Blackbox

DB2.8 Functional

DB2.9 Unit

DB3.1 **Database Test 3**

DB3.2 The purpose of this test is to show that queries for player game history returns in a timely manner.

DB3.3 The database should return the previous 100 games of a user in less than 3 seconds.

DB3.4 Input: SQL Query

DB3.5 Output: List of games and Time in seconds

DB3.6 Normal

DB3.7 Whitebox

DB3.8 Performance

DB3.9 Unit

FS1.1 **Full System Test 1**

FS1.2 The purpose of this test is to ensure that the database can be accessed from the web application.

FS1.3 The web application should be able to make a request and receive data that is currently in the database for both user info and game history.

FS1.4 Input: Web request

FS1.5 Output: List of games or List of user info

FS1.6 Normal

FS1.7 Blackbox

FS1.8 Functional

FS1.9 Integration

III. Test Case Matrix

	Normal/ Abnormal	Blackbox/ Whitebox	Functional/ Performance	Unit/ Integration
CE1	Normal	Blackbox	Functional	Unit
CE2	Normal	Blackbox	Functional	Unit
CE3	Normal & Abnormal	Whitebox	Performance	Unit
CE4	Normal	Whitebox	Performance	Unit
R1	Normal	Blackbox	Functional	Unit
R2	Normal	Blackbox	Functional	Unit
R3	Normal	Blackbox	Functional	Unit
R4	Normal	Blackbox	Functional	Unit
DB1	Normal	Blackbox	Functional	Unit
DB2	Normal	Blackbox	Functional	Unit
DB3	Normal	Whitebox	Performance	Unit
FS1	Normal	Whitebox	Functional	Integration