**Roy’s Chess Game**

**פרויקט גמר של סדנה בתכנות מתקדם בשפת Java‏–**

**מגיש: רועי יצקן**

1. **תיאור מערכתי**

הפרויקט הוא משחק שחמט, המאפשר משחק הן בין 2 משתתפים והן מול המחשב.

**מאפיינים של הפרוייקט:**

**מודל שרת–לקוח**: הפרויקט בנוי בארכיטקטורת שרת–לקוח. השרת הוא "המתכלל" – כל מעבר מצב (התחברות, כניסה למשחק..) כלשהו אצל הלקוח מתבצע אחרי אישורו והוא אחראי ללוגיקה מאחורי משחק השחמט. צד הלקוח יודע רק להדפיס את הלוח שקיבל מהשרת.

**תקשורת**: לשם תקשורת השתמשתי בRMI השרת שולח את הRemote Object שלו לכלל הלקוחות, והם שולחים לשרת את הRemote Object שלהם. כך מתבצעת תקשורת דו כיוונית , כאשר כל צד רוצה בכך.

**Framework**: השתמשתי בSpring Boot.

**Database**: מסד הנתונים, הכולל פרטי משתמשים ואת הצאטים בינהם נשמר גם אחרי סגירת השרת. לשם כך השתמשתי בH2 (database פשוט שהוא embedded), וJPA הניתמך ע"י Spring Boot, הדואג לכל "העבודה השחורה" מאחורי הקלעים.

**GUI**: הGUI הוא JavaFX, השתמשתי בספריית materialfx, המציעה רכיבי Javafx במראה מודרני, וקובץ דמו שהיווה שלד לקוד שלי.

**ספריית שחמט:** ספריית השחמט, שכתבתי בעצמי שנמצאת בצד שרת, משרתת 2 מטרות עיקריות: הכרעה של סיום המשחק והכרעת המסעים החוקיים שניתן לבצע במצב מסוים. ספריית השחמט תתמוך בכלל החוקים במשחק (מלבד חוק ה–50 מסעים), כמו – הצרחה, הכאה דרך הילוכו, הכתרה, קידום חייל ב2 מסעים ופט.

**Stockfish:** Stockfish הינו מנוע שחמט החזק ביותר כיום. קובץ EXE שלו מצורף לפרויקט, וכתבתי סקריפט בפייתון אשר מכריע את המסע הכדאי ביותר במצב מסוים. הסקריפט מסתמך בספריית קוד בפייתון המתקשרת עם אותו קובץ EXE, אשר לא קיימת בJAVA.

**צאט:** המשחק מכיל צ'אט, כך שכל משתמש יוכל לשלוח הודעות למשתמשים אחרים אשר מחוברים כעת. ניתן לגשת לצאט מכל מסך (הוא נמצא במסך ה"super"(.

**תיאוריה:**

מקובל לסווג מהלכים במשחק שחמט לשלושה קטגוריות –

* מהלך חוקי – מותר לשחקן לבצע אותו.
* מהלך פסיאדו–חוקי – מהלך חוקי ע"פ העקרונות הבסיסיים של המשחק, חוץ מהחוק המחייב את הגנתו של המלך אשר לא נבדק.
* מהלך לא חוקי.

Forsyth–Edwards Notation **(FEN)**– ייצוג של מצב של לוח שחמט. בין היתר שומר מידע על מיקום הכלים בלוח, הזכות של כל צד להצרחה, מידע לגבי הכאה דרך הילוכו

**תכנון הפרויקט:**

**צד לקוח:**

בלקוח שלושה אלמנטים ראשיים: sharedCode, ספריית הקוד המשותפת, SuperController שהוא הבקר של דף העל, GameController האחראי לניהול משחקים וPlayController, הבקר של דף המשחק.

* **sharedCode** – ספרייה קוד משותפת לשרת וללקוח.
  + Chat – צאט בין שני משתמשים.
  + Color – שחור/לבן, צבע של שחקן במשחק שח
  + GameEntry – entry לטבלת המשחקים בדף הראשון.
  + GamePiece – מייצג כלי במשחק.
  + King/Knight/Pawn/Queen/Rook/Bishop – כלים במשחק, היורשים מGamePiece.
  + ClientPiece – מייצג כלי במשבצת, מחלקה נטולת לוגיקת שחמט המיועדת ללקוח.
  + GameResult – תוצאה של משחק שח
  + GameUpdate – עדכון כל הנתונים שהלקוח צריך לדעת לגבי משחק כלשהו. (הלוח, שמות משתתפים...)
  + PieceEncoding – זהה לGamePiece, אבל לשימושים אבסטרקטים.
  + PieceMovement – מייצג תנועה של כלי.
  + Square – מייצג משבצת בלוח.
* **Controllers** – בקרים של התכנית. מכיל את הבקרים הסטנדרטים של כל קובץ FXML ל4 הסצנות במשחק וסצנת ה”super”.
  + SuperController – controller של מסך העל
  + AboutPageController – הcontroller של מסך המידע.
  + HomePageController – הcontroller של מסך הבית.
  + InitializerPageController – הcontroller של מסך אתחול משחק חדש
  + PlayController – controller של מסך המשחק שחמט
  + Initializer – עוזר לאתחול הSuperController, ומאחסן קוד לא אסתטי.
  + GameController – שולט בעיניינים הרלוונטים למשחק שחמט, יחד עם הPlayController.
* **Communication** – מחלקה ServerActionInClient מהווה הRemote Object של הלקוח, ומחלקת Helper המכילה מתודות של RMI.
* **Board** – BoardCell מהווה תא בלוח השחמט הוויזואלי, שהוא מיוצג ע"י מחלקת BoardPane.
* **Application** – המחלקה המכילה את המתודה main של צד הלקוח, ומאתחלת את spring boot.
* **Data** – מחלקה המאחסנת מידע המשותף לכלל המחלקות האחרות בצד שרת.

**צד שרת:**

בשרת שני מחלקות ראשיות: ConnectionManager, שאחראי על מענה לפניות לשרת וchesslib, ספריית השחמט האחראית ללוגיקת המשחק.

* **chesslib**: ספריית השחמט, המנותקת מאשר התיקיות.
  + **Game**: ה"לב" של הספרייה. 3 המחלקות הראשונות פעולות בקשר הדוק.
    - Board – מחלקה המייצגת לוח שחמט מצד אחד ומהווה "שופט" הבודק חוקיות של מסעים מצד שני ושומרת על תקינות פנימית. מכילה הרבה "moving parts" בתוכה ולכן מועדת לפורענות. המחלקה הגבוהה ביותר בהיררכיה.
    - BoardData – נמצאת בתוך כל Board, ושומרת את המידע הנאסף עד כה על המשחק, אשר לא נשמר בFEN.
    - PieceStorage – נמצאת בתוך כל Board, ופיזית מאכסנת את כל הכלים על הלוח.
    - FenCode – ייצוג Fen ללוח. (מסוכן כי לא ניתן להשלים את BoardData).
  + **Helper**: מכילה מחלקות עזר.
    - CastlingRight – Enum המייצגת זכות של שחקן מסוים בלוח לבצע הצרחה (גדולה, קטנה, שניהם, אף אחד).
    - MoveType – כל מסע ניתן לשייך לסוג אחד בMoveType.
    - PlayerMovement – מייצג מסע על הלוח, קיימת מחלקה נוספת המכילה פשוט גם MoveType.
  + **Local**: מכילה מחלקות הרלוונטיות להרצה מקומית של משחק.
    - LocalGame – מחלקת שלד המאפשרת לנו להריץ משחק שח באופנים שונים.
    - PreRunGame – מחלקה היורשת מLocalGame, ומאפשרת הרצת משחק כלשהו מראש, כאשר המשתתפים בו הם מטיפוס PreRunPlayer.
    - StockFishGame – מחלקה היורשת מLocalGame, מאפשרת הרצת משחק כאשר שני השחקנים הם Stockfish, מטיפוס StockfishPlayer.
    - Player – Interface המייצג שחקן שממנו יורשים שני מחלקות השחקנים StockfFishPlayer ו PreRunPlayer.
  + **Mainlogic**: מחלקת הלוגיקה של הספרייה. פועלת עבור תקיית Game.
    - BoardMaps – המחלקה הכי נמוכה בהיררכיה בתיקיה. אחראית על יצירת moveMap, threatMap, eatMap.
    - Logic – מחלקת שאילתות סטטיות לגבי חוקיות מהלכים. MoveGenerator וBoard נעזרים בה.
    - MoveGenerator – אחראית על יצירת מהלכים חוקיים לכל Board, תוך היעזרות בשאר המחלקות בתיקייה.
    - Stockfish – מחלקה האחראית על התקשרות עם קובץ הפייתון – הרצתו, שליחת קלט אליו וקבלת פלט.
  + **Result**: מכילה מחלקות שהם תוצאות של משחקים. המחלקה שימושית לאימות הספרייה אל מול ספריות מקובלות.
* **Database**: מכיל שתי מחלקות משניות, הרלוונטיות למסד הנתונים של השרת.
* **Communication**:
  + - עמוד השדרה של השרת, האחראי לקבלת הפניות מהלקוחות. הראשי הוא ConnectionManager, שהוא גם הRemote Object, הנעזר בעזרת שאר המחלקות. חיונותו נובעת מהעובדה שהשרת בעיקר מגיב לפניות ולא יוזם בעצמו.
    - GameController – הבקר של כל המשחקים. פיזית שומר את המשחקים ואמון על פניות הקשורות אליהם.
    - UserController – שומר את כל המשתמשים המחוברים, ואמון על פניות אליהם (בעזרת Remote Object).
    - GameState – בקר של משחק מסוים, יש שני סוגי משחקים לכן שתי מחלקות.
* **Resources/chess**: מכיל את המשאבים הרלוונטים לספריית השחמט. מכיל את הקבצים המשמשים את Stockfish וקבצי המשחק המשמשים לאימות הספרייה.
* **Test**: מכיל את מחלקת MainTest, מחלקה קטנה שאחראית להרצת אימות ספריית השחמט.

1. **מסכים של צד הלקוח:**

**הGUI מכיל 5 מסכים:**

* מסך ה–super (מסך העל): מסך זה הוא למעשה הGUI, כאשר כל שאר המסכים מוכפפים בתוכו. במסך זה נמצא התפריט של 3 מסכים והצ'אט.
* מסך הAbout: מסך הנמצא בתפריט, מכיל רקע על הפרויקט.
* מסך הHomepage: מסך זה נמצא בתפריט ומכיל את טבלת המשחקים, שממנה משתמש יכול להצטרף למשחקים שמשתמשים אחרים פתחו.
* מסך ה–Initializer: מסך זה נמצא בתפריט ומתוכו אפשר ליצור משחק מול המחשב או מול משתמש אחר.
* מסך ה–Play: מסך זה לא נמצא בתפריט, ומהווה מסך המשחק שבתוכו מתרחשים משחקי השחמט.

1. **מסמך מנקודת מבטו של המשתמש:**

**טרום–משחק:**

המשתמש מזין את פרטי הכניסה (משתמש וסיסמא) בConsole. הפרטים נשלחים לאימות אצל השרת, שמחזיר כן או מורה למשתמש להזין בשנית.

לאחר אישור השרת, מסך העל נפתח. המשתמש לתקשר עם משתמשים אחרים בצא'ט (גם במהלך משחק), ולדפדף בתפריט מצד שמאל.

כדי לשחק, המשתמש יבחר בשלוש אפשרויות:

**הצטרפות למשחק קיים** – המשתמש יוכל לבחור במסך הראשון בטבלה משחקים שמשתמשים אחרים פתחו, ומחכים למצטרף. הוא יוכל להצטרף לאחד מהם, ואז ישלח למסך המשחק, ומשחק זה יורד מהטבלה.

**פתיחת משחק מול משתמש אחר** – במסך השני, המשתמש יוכל ליצור משחק מול משתמש אחר. לאחר פתיחתו, ימתין המשתמש למישהו אשר יצטרך למשחק (בעזרת האופציה לעיל).

**פתיחת משחק מול המחשב** – במסך השני, המשתמש יוכל ליצור משחק מול המחשב. לאחר פתיחתו המשחק יתחיל במיידית, וכמובן שלא יוצע לבחירה בטבלת המשחקים.

**מהלך משחק:**

המשחק יתנהל כמו שנהוג במשחקי שחמט – בעת לחיצה על משבצת, יוצג למשתמש המסעים החוקיים שיוכל לבצע מאותה המשבצת (בהנחה שזה תורו) – המשבצות שכלי שנבחר יכול לנוע אליהם. לאחר הלחיצה הראשונה, אם ילחץ על משבצת שיכול לנוע אליה יתבצע המסע. אך אם ילחץ על משבצת שלא יכול לנוע אליה, יתאפסו הלחיצות.

בנוסף, המשתמש יוכל להציע תיקו בכפוף לאישורו של היריב או להיכנע ולצאת מהמשחק למסך Homepage.

1. **מנקודת מבטו של השרת:**

לאחר העלאת השרת (MainServer), ניתן להריץ פקודות המחזירות מידע על מצב השרת:

games – מחזירה את שמות המשחקים הפעילים

activeUsers – שמות המשתמשים המחוברים

savedUsers – שמות המשתמשים שנשמרו במסד נתונים

allChats – מחזיר את כל הצאטים של מכל המשתמשים.

בצד השרת, ניתן להריץ בדיקות אמיתות לchesslib במחלקה MainTest. ניתן גם להריץ משחק Stockfish, שבו המנוע משחק מול עצמו.

1. **עמידות ונכונות:**

**מבחינת השרת:**

העקרון החשוב בעיניי הוא שמירה של תקינות בצד השרת. כל פנייה של הלקוח אל השרת מלווה בשמו ואם זה רלוונטי שם המשחק בשביל להימנע משגיאות בלקוח להגיע לשרת. ישנם שגיאות רבות שיכולות לצוץ הקשורות בעיקר לתקשורת, בין ביתר ממקביליות. השרת מתקשר עם הלקוח כל זמן קבוע, ומשתמשים לא זמינים מפונים, והמשחקים שבהם נמצאים נמחקים.

**מבחינת ספריית השחמט:**

מכיוון שכתבתי את הספרייה בעצמי יש סיכוי לטעויות. לשם כך השוואתי את הספרייה מול אחת סטנדרטית של JAVA אשר בוצעה על אלפי משחקים. עבור כל משחק, כל מצב של המשחק נבחן בנפרד – הלוח עצמו (המיוצג כFEN) ורשימת המסעים החוקיים.

עדיין יתכן שעשיתי טעות בפונקציות השוואה עצמם, ולא בחנתי את כלל המקרים.

**מצבים שנשקלו/נבדקו:**

* יציאה במפתיע של שחקן תמחק אותו מהמערכת ואם היה בתוך משחק, תסיים את המשחק.
* הזנה של פרטים שגויים של משתמש שמור.
* כניסה תקנית כמשתמש כלשהו, כאשר נמצא כבר מישהו מחובר באותו השם.
* לא ניתן לבצע מסעים לא חוקיים במשחק שחמט.
* שם משתמש/סיסמא ארוכים/קצרים מידי.
* משתמש ישתמש בrebind, במקום bind למקרה שמשתמש שהתנתק לא ביצע unbind אוטומטית.
* יצירת משחק עם שם של משחק קיים או שם קצר/ארוך מידי.
* בחירה לא של רק משחק אחד בטבלה
* מצב של פט גורר תיקו.
* בחירה כל הכלים האפשריים להכתרה (לא רק מלכה).

1. **העזרות בספריות (הלא קונבנציונליות):**

MaterialFX – ספריית הJAVAFX

<https://github.com/palexdev/MaterialFX/wiki>

chesslib של bhlangonijr – ספריית שחמט שנעזרתי בה לולידציית הספריה שלי.

<https://github.com/bhlangonijr/chesslib>

py stockfish – ספריה בפייתון המתקשרת עם Stockfish.

<https://pypi.org/project/stockfish/>

Jess – פרויקט שחמט, שנעזרתי בו בעיקר בשביל BoardPane וBoardCell.

<https://github.com/gameofjess/Jess>