מסמך תכנון

שרת משחקים – "מלכה לב"

מגיש: שלומי חבר

ת.ז: 204096648

תוכן עניינים

[תיאור כללי: 3](#_Toc19699717)

[העלאת המערכת: 3](#_Toc19699718)

[בסיס הנתונים: 3](#_Toc19699719)

[תרשים כללי של המערכת: 5](#_Toc19699720)

[ספריות JAVA חיצוניות: 6](#_Toc19699721)

[רשימת החבילות שבהן עשינו שימוש: 6](#_Toc19699722)

[חבילות פנימיות: 7](#_Toc19699723)

[מבנה מחלקות: 7](#_Toc19699724)

[חבילת Logic: 7](#_Toc19699725)

[חבילת Controller: 7](#_Toc19699726)

[חבילת Model: 8](#_Toc19699727)

[חבילת Repository: 8](#_Toc19699728)

[מחלקות נוספות: 8](#_Toc19699729)

[UML: 9](#_Toc19699730)

[פרויקט Client: 12](#_Toc19699731)

[Css: 12](#_Toc19699732)

[Js: 12](#_Toc19699733)

[Assets: 12](#_Toc19699734)

# תיאור כללי:

המערכת המוצגת היא שרת משחקים המסוגל להריץ את המשחק: "מלכה לב" (חוקי המשחק מתוארים במסמך התיאור).

טכנולוגיות בשימוש במערכת:

* Mysql server – משמש כבסיס המידע של המערכת.
* JDBC – ממשק התקשורת עם בסיס הנתונים.
* JPA – ממשק תקשורת עם מס"ד הנתונים המאפשר מיפוי של אובייקטים לטבלאות בבסיס הנתונים.
* Spring – אוסף של ספריות שמאפשרות הרצה ובניה של מערכות WEB.
* JAVA Web service – ממשק לקבלת מידע מהאפליקציה בפורמט JSON.
* HTML/CSS – טכנולוגיות לבניה ועיצוב של דפי אינטרנט.
* React – framework javascript לבניה של Single page application.
* Web sockets – טכנולוגיה המאפשרת שליחת הודעות מהשרת ללקוח ביוזמת השרת.

# העלאת המערכת:

התהליך מתואר בקובץ ה- README.

נגישות למערכת:

המערכת נגישה דרך דפדפן האינטרנט של המחשב או הפלאפון, על ידי הכתובת הבאה: <http://localhost:3000/>.

# בסיס הנתונים:

מערכת זו משתמשת בבסיס נתונים מסוג: mysql, גישה למס"ד נעשית באמצעות Spring Jpa שמבצע חיבור בין האפליקציה למערכת בעזרת הכתובת הבאה: jdbc:mysql://localhost:3306.

שם מס"ד הנתונים הוא: db\_queen\_of\_heart

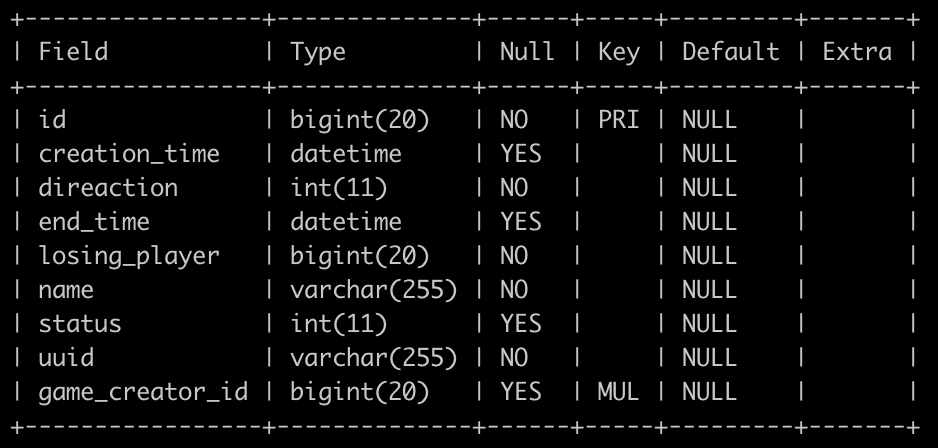
טבלאות:

|  |  |
| --- | --- |
| שם הטבלה | תיאור |
| Games | טבלה המכילה את כל המשחקים שמנוהלים במערכת |
| Players | טבלה המכילה את כל השחקנים ששיחקו במשחקים שונים, ומקושת עם מפתח של משחק לטבלה Games |
| Actions | כל שורה בטבלה הזו מקושרת למשחק אחד, ומכילה את הפקודה שכל השחקנים צריכים להישמע לה, לדוגמה: כך קלף, שנה כיוון לשחקן הבא וכו... |
| game\_play\_history | היסטוריה של המשחק, מכילה את רצף שליפת הקלפים של המשחק, כל שורה מכילה את השחקן ששלף את הקלף, את הקלף שנשלף על ידי השחקן המיקום שלו על המגרש.  מידע זה מאפשר לנו לשחזר את מצב המשחק במקרה הצורך. |

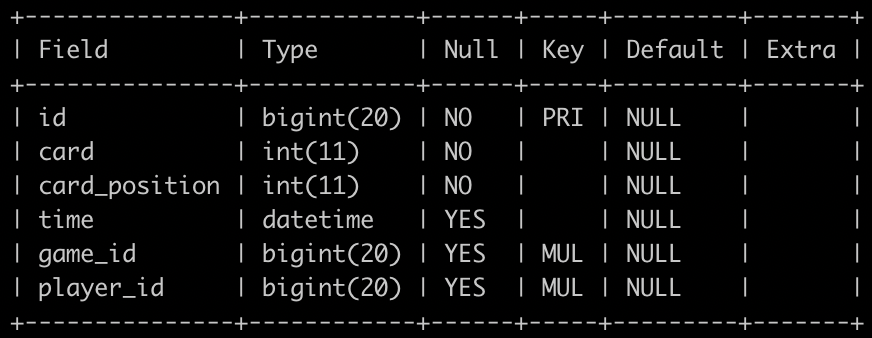
כל הטבלאות מכילות שדות ראשיים שמיוצרים אוטומטית על ידי המס"ד ו- JPA, לטבלאות Games, Players ישנו שדה מיוחד שמוגדר להיות כמזהה יחודיי עבור אותה שורה. כזה שמוגרל בצורה רנדומלית ולא ניתן לשחזר אותו הטיפוס של שדה זה הוא: uuid.

מבנה הטבלאות הוא:

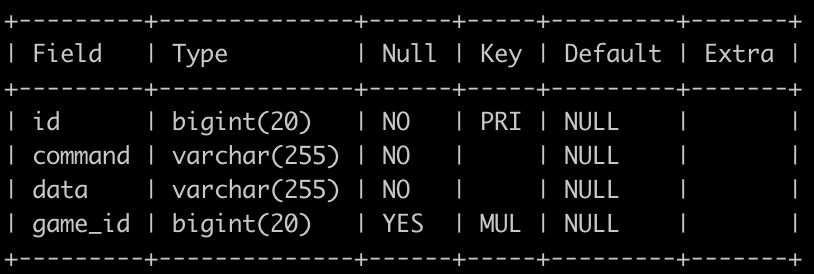
Game:



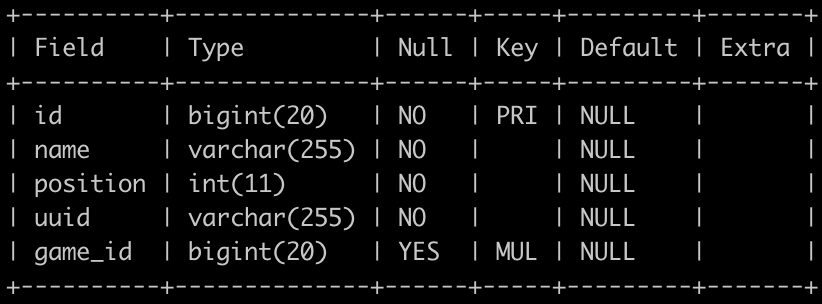
Game\_play\_history:



Actions:



Players:



# תרשים כללי של המערכת:

Logic

Controllers

Business layer

Data access layer

JPA - hibernate

Model

Client

SPA – (React)

Communication layer

Web service

Web socket

DB

# ספריות JAVA חיצוניות:

השימוש בחבילות חיצוניות מוגדר בקובץ שנקרא: pom.xml קובץ זה מגדיר את כל התלויות של הפרויקט בפורמט של Maven שהוא מנהל החבילות שבו הפרויקט משתמש.

החבילות שמוגדרות בקובץ מתוארות באופן הבא:

<dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-web</artifactId>  
</dependency>

לכל תלות יש את שם החבילה שלה: artifact-id ואת שם המפיץ או חבילת ההפצה עצמה: group-id

לפעמים גם מגדירים את הגרסה הרצויה של החבילה.

## רשימת החבילות שבהן עשינו שימוש:

חבילות WEB:

* spring-boot-starter-web
* spring-boot-starter-data-rest
* spring-boot-starter-thymeleaf
* gson
* json-path

חבילות תקשורת:

* spring-boot-starter-websocket

קישוריות DB:

* h2
* spring-boot-starter-data-jpa
* mysql-connector-java
* hibernate-core
* hibernate-entitymanager

תלויות פנימיות של המערכת:

* jackson-mapper-asl
* jackson-core-asl

חבילות פיתוח:

* spring-boot-devtools

# חבילות פנימיות:

לצורך מימוש הלוגיקה של האפליקציה, בנינו מספר מחלקות פנימיות שעוזרות במימוש החלקים הנדרשים.

## מבנה מחלקות:

### חבילת Logic:

חבילה המכילה את כל המחלקות שקשורות בניהול המשחק בצד השרת.

|  |  |
| --- | --- |
| שם המחלקה | תיאור |
| Actions | ישנן מחלקות עבור הפעולות השונות ששחקנים מבצעים תחת תת החבילה Actions |
| ActionManager | מחלקה לניהול הפעולות במשחק |
| Card, CardType, Deck | ממשות את הפעולות שניתן לבצע על קלפים בחבילת המשחק |
| Rules | מחלקה המנהלת את כל חוקי המשחק בהינתן המצב של המשחק והקלף האחרון שנשלף |

### חבילת Controller:

חבילה המנהלת את כל התגובות שהשרת מחזיר לשחקן במהלך המשחק.

|  |  |
| --- | --- |
| שם המחלקה | תיאור |
| CardController | בקר שאחראי על פעולות שקשורות בניהול הקלפים במהלך המשחק:   * הגרלת קלף עבור שחקן |
| GameController | בקר שאחראי על פעולות שקשורות בניהול המשחק:   * יצירת משחק * התחלת משחק * קבלת פרטים על המחשק * קבלת רשימת המשחקים של השחקן |
| PlayerController | בקר שאחראי על פעולות שקשורות בניהול שחקנים:   * יצירת שחקן * קבלת פרטים על השחקן |
| Response | מחלקה המגדירה את מבנה התגובה שחוזר מהשרת (Web service) |

### חבילת Model:

חבילה המנהלת את כל האובייקטים שממופים לטבלאות במשחק.

|  |  |
| --- | --- |
| שם המחלקה | תיאור |
| כל המחלקות | מיפוי של האוביקטים למבנה של הטבלאות בסיס הנתונים שימוש בטכנולוגית JPA ו- hibernate |

### חבילת Repository:

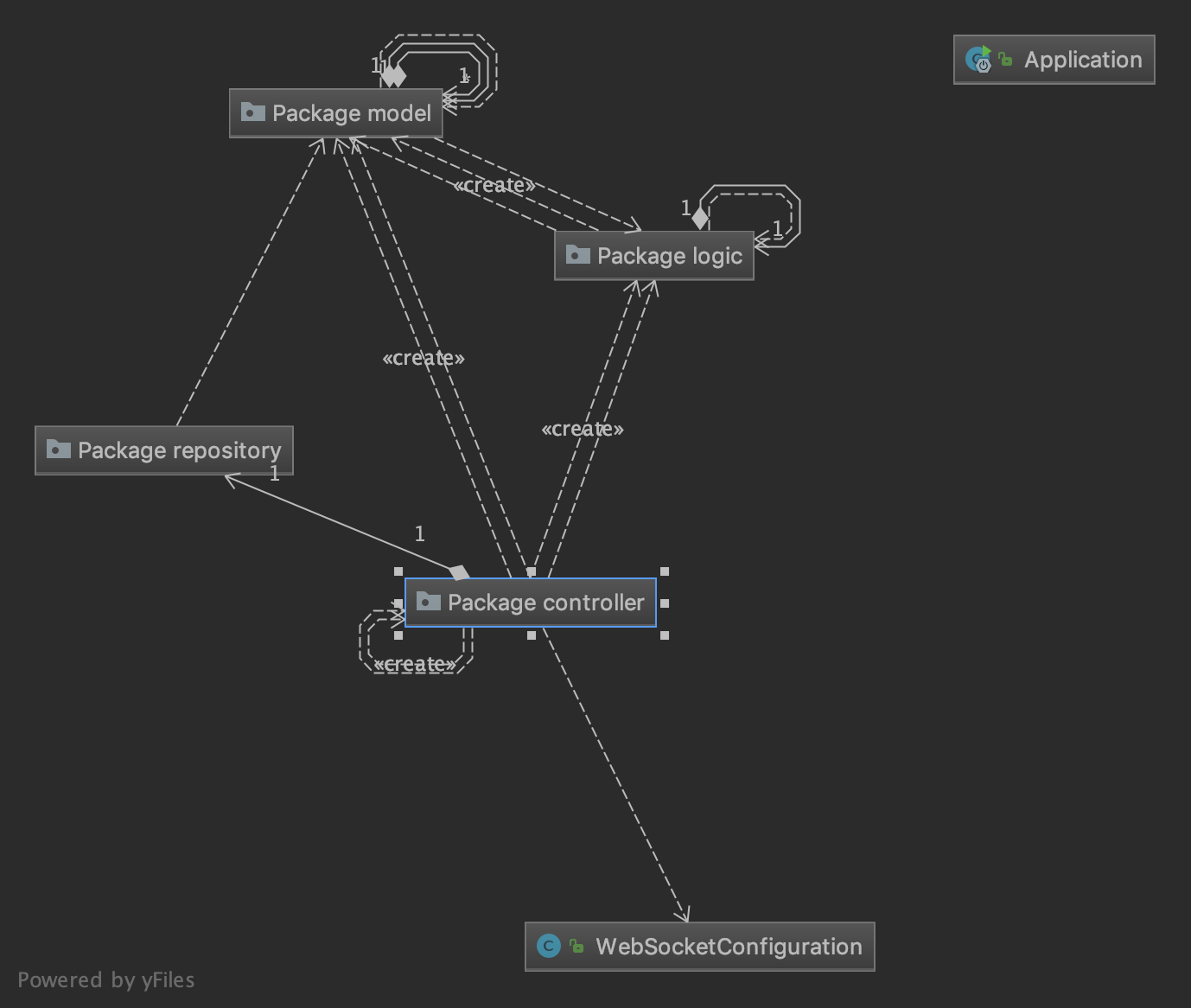
חבילה המכילה את ממשקים שמאפשרים כתיבה, קריאה וניהול של אוביקטים במס"ד הנתונים, כל ממשק מרחיב את אוסף הפעולות הבסיסי של הממשק: CrudRepository, שמגיע מהחבילה: org.springframework.data.repository.

### מחלקות נוספות:

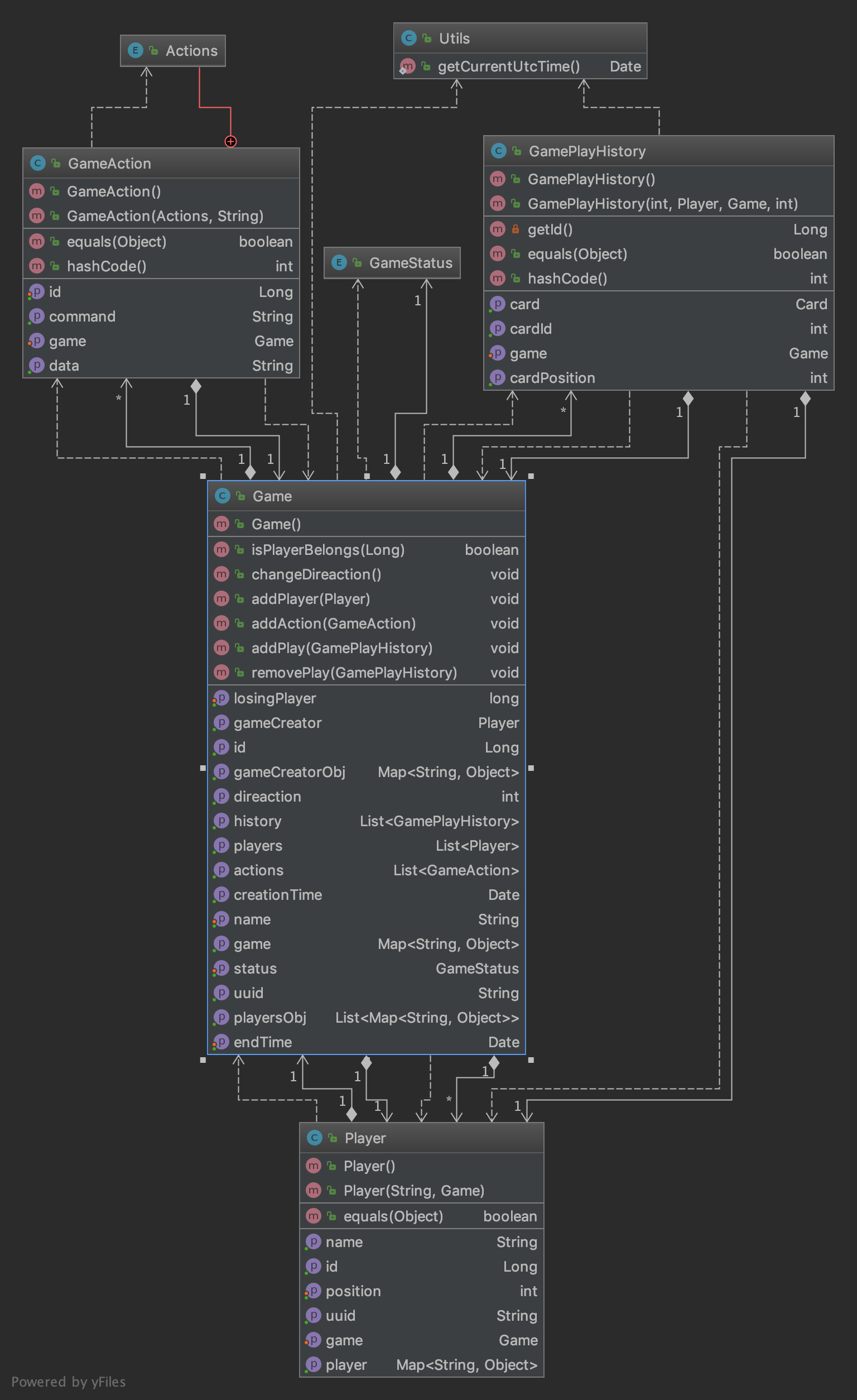
|  |  |
| --- | --- |
| שם המחלקה | תיאור |
| Utils | מחלקה המכילה אוסף פעולות שיש להן שימוש רב פעמי באפליקציה, על ידי מחלקות שונות |
| Application | מחלקה שמוגדרת כנקודת הגישה של האפליקציה, היא זו שאחראית על הרמת השרת |
| WebSocketConfiguration | הגדרה של תשתית ה- webscokets, שאנו משתמשים בה. |

# UML:

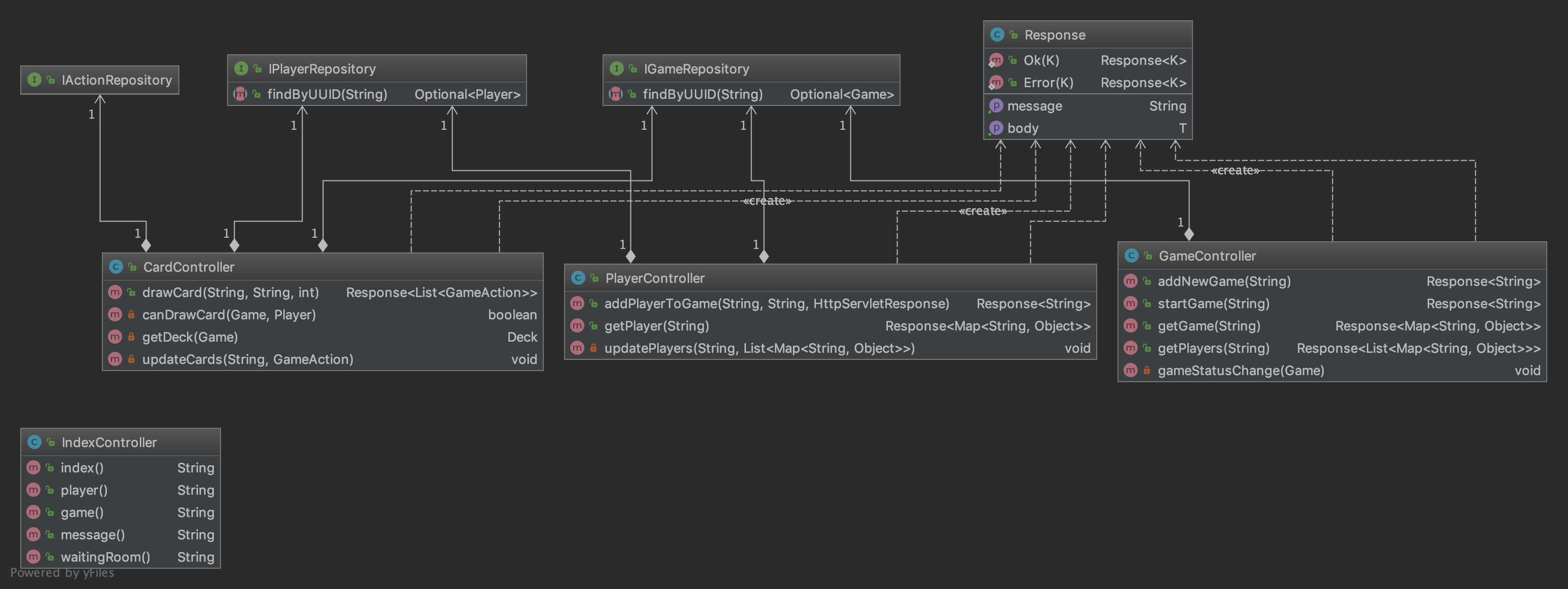
מבנה כללי של תלויות בין חבילות:



מבנה מחלקות בחבילה של Model:



מבנה מחלקות בין חבילות Repository ו- Controller



# פרויקט Client:

בחבילה זו מוגדרים כל הקבצים שמונגשים עבור הלקוח של האפליקציה.

## Css:

תיקיה המכילה את כל הגדרות העיצוב של האפליקציה

## Js:

תיקיה המכילה את כל הקוד שרץ בדפדפן של הלקוח, קוד זה אחראי על כל לוגיקת המשחק שמתנהלת בדפדפן.

בעקבות השימוש בספריה בשם react, כל קובץ מכיל רכיב (component) נפרד שאחראי על חלק אחד באפליקציה, כל רכיב אחראי על:

* ניהול המצב שלו (state)
* ניהול התצוגה שלו, רכיב אחראי על render המידע שיש בו בתוספת העיצוב שמוגדר עבורו
* כל ההתנהלות מול השרת, שליחת בקשות שינוי העיצוב או המצב בהתאם לתגובה

## Assets:

Index.html

העמוד הראשי שאחראי על הצגה של כל תוכן האתר.

Images

מכיל את כל אוסף התמונות שמונגשות באפליקציה (הקלפים).

כל הקבצים מנוהלים על ידי תוסף שנקרא webpack, תוסף זה אחראי על איחוד כל קובצי ה- js לקובץ אחד מינימלי למטרות שיפור יעילות האפליקציה.