

# מסמך תכנון המתאר את מבנה המערכת, תיאור המחלקות השונות והקשרים ביניהם

אלמוג סיטון

[almogsiton@gmail.com](mailto:almogsiton@gmail.com)

205501901

תוכן עניינים

פרק 1 - מבוא עמוד 3

פרק 2 - ארכיטקטורת המערכת עמודים 4-5

פרק 3 – מבנה הנתונים עמודים 6-11

פרק 4 – תיאור המחלקות והקשרים ביניהן עמודים 12-14

פרק 5 – קבצי XHTML והקשר למחלקות עמודים 15-16

פרק 6 – זרימת מידע בין שכבות עמוד 17

פרק 7 – היבטים מערכתיים עמוד 18

## פרק 1 – מבוא

### 1. מטרת המערכת:

מערכת **Almoviland** נבנתה לצורך ניהול ספריית סרטים דיגיטלית, ומיועדת לאפשר למשתמשים לבצע מגוון פעולות באופן מקוון, כגון: השאלת סרטים, צפייה בפרטים, ניהול ביקורות ודירוגים, והחזרת סרטים. בנוסף, מנהלי המערכת יכולים לנהל את מלאי הסרטים, המשתמשים, הקטגוריות, ולפקח על השאלות. המערכת פועלת בסביבה מקומית (localhost) על שרת GlassFish .

### 2. מטרות הפרויקט:

- פיתוח מערכת מודולרית לניהול ספריית סרטים בשפת Java.
- יישום עקרונות תכנות מונחה עצמים ומVC .
- שימוש בטכנולוגיות Web מתקדמות כגון CSS, XHTML, JSF ו- RESTful API.
- התממשקות למסד נתונים (Apache Derby) לצורך ניהול מידע על משתמשים, סרטים, השאלות וביקורות.
- הבטחת חוויית משתמש חלקה וידידותית תוך הפרדה בין הרשאות משתמשים.

### 3. טכנולוגיות וכלים:

- המערכת פותחה בסביבת **Apache NetBeans 23** עם שימוש בטכנולוגיות הבאות:
- **Jakarta Java EE** פלטפורמה ליישומי Web .
  - **JSF (JavaServer Faces)** תשתית ליצירת ממשקי משתמש מבוססי קומפוננטות.
  - **JDBC** גישה ישירה לבסיס הנתונים.
  - **JavaScript + CSS + XHTML** לבניית ממשק משתמש רספונסיבי ודינמי.
  - **RESTful Web Services (JAX-RS)** חשיפת מידע בפורמט JSON .
  - **GlassFish Server 7** שרת לפריסה מקומית של האפליקציה.
  - **Apache Derby** מסד נתונים לניהול מידע.

## פרק 2 – ארכיטקטורת המערכת

### 1. סקירה כללית:

מערכת Almoviland מבוססת על ארכיטקטורת MVC (Model-View-Controller), המפרידה בין שכבת הנתונים (Model), שכבת הלוגיקה העסקית והבקרה (Controller) ושכבת התצוגה (View). מבנה זה מאפשר הפרדה ברורה בין תחומי אחריות, תחזוקה קלה יותר של הקוד, והרחבה נוחה של רכיבים קיימים. המערכת מיועדת לפעול בסביבה מקומית, תוך שימוש בשרת GlassFish ובמסד נתונים מסוג Apache Derby כאשר כל פעולה מתבצעת על ידי מעבר מסודר בין שכבות המערכת.

### 2. מודול לקוח (Client)

מודול הלקוח כולל את הממשק הגרפי של המשתמש הרגיל ומורכב מקבצי XHTML המעוצבים בעזרת CSS ונתמכים ב־ JavaScript. במודול זה המשתמש יכול להירשם למערכת, להתחבר, לחפש סרטים לפי שם או קטגוריה, להשאיל סרטים, להחזיר סרטים, לדווח על אובדן, להוסיף ביקורות ולדרג סרטים. בנוסף, לכל משתמש קיים עמוד פרופיל אישי שבו ניתן לצפות בהיסטוריית ההשאלות ולבצע פעולות נוספות כמו חזרה או דיווח. ממשק המשתמש מותאם לתפקידו של המשתמש המחובר ומציג רק את הרכיבים הרלוונטיים עבורו.

### 3. מודול מנהל (Admin)

מודול המנהל מיועד למשתמשים בעלי תפקיד ADMIN ומספק להם ממשק ניהול עשיר. באמצעותו ניתן להוסיף, לערוך ולמחוק סרטים, לנהל קטגוריות כולל תמונות, למחוק ביקורות של משתמשים אחרים, לנהל את רשימת המשתמשים (כולל שינוי תפקידים ומחיקה), ולאשר דיווחי אובדן. למנהלים יש גישה לעמודי ניהול ייעודיים כגון `moviesManagement.xhtml`, `categoriesManagement.xhtml`, `usersManagement.xhtml` ו־`lossReports.xhtml` כל רכיבי הניהול מוצגים רק כאשר המשתמש המחובר הוא בעל הרשאה מתאימה.

### 4. מודול שרת (Server-Side Logic)

מודול השרת אחראי על עיבוד הלוגיקה העסקית של המערכת. הוא כולל מחלקות Java מסוג Managed Beans אשר שולטות בהתנהגות הדינמית של המסכים ומתווכות בין התצוגה למסד הנתונים. פעולות על הנתונים מתבצעות באמצעות מחלקות DAO (Data Access Object) אשר משתמשות ב־ PreparedStatement לצורך גישה בטוחה לבסיס הנתונים. בנוסף, המערכת כוללת שירות REST מסוג JAX-RS (MovieRestService) המאפשר שליפת נתונים בפורמט JSON דרך כתובת מוגדרת. כל החיבורים, החישובים, הוולידציות, והבדיקות העסקיות מנוהלים במודול זה בצורה מבוצרת, ברורה, ומודולרית.

## 5. שכבות הליבה ויחסי הגומלין:

המערכת מקפידה על הפרדה מוחלטת בין השכבות. שכבת ה־ View בנויה בקבצי XHTML המתקשרים עם שכבת ה־ Beans דרך EL (Expression Language) של JSF. כל Bean מחזיק את הנתונים והפעולות עבור דף מסוים, ומתקשר עם מחלקות DAO שמבצעות את הפעולות על מסד הנתונים. שכבת ה־ Model כוללת את מחלקות הנתונים כמו User, Movie ו־ Borrowing. שכבת ה־ Controller מיוצגת בעיקר על ידי מחלקות Beans כמו MovieBean או UserBean אשר שולחות קריאות ומבצעות לוגיקה בהתאם לפעולות המשתמש. שירות ה־ REST מהווה שכבה חיצונית נפרדת לצרכים של אינטגרציה עתידית.

### פרק 3 – מבנה הנתונים

#### טבלת USERS

טבלה זו מכילה את כל המשתמשים הרשומים במערכת – הן משתמשים רגילים והן מנהלים.  
כל שורה מייצגת משתמש ומכילה את השדות הבאים:

USERS
USER_ID מזהה ייחודי לכל משתמש (מפתח ראשי)
NAME שם מלא של המשתמש
EMAIL כתובת הדוא"ל (ייחודית)
PASSWORD הסיסמה המוצפנת של המשתמש
ROLE תפקיד המשתמש
LIMIT_BORROW_MAX מספר הסרטים המרבי שמשתמש יכול להשאיל
DATE_REGISTRATION תאריך ההרשמה למערכת

טבלת MOVIES

טבלה זו מייצגת את כל הסרטים במערכת.  
כל שורה מייצגת סרט בודד, ומכילה את השדות הבאים:

MOVIES
MOVIE_ID מזהה ייחודי לכל סרט (מפתח ראשי)
TITLE שם הסרט
DESCRIPTION תקציר תוכן הסרט
QUANTITY סך כל העותקים שקיימים למלאי
COPIES_AVAILABLE מספר העותקים הזמינים להשאלה כרגע
DATE_RELEASE תאריך יציאת הסרט
POSTER_IMAGE תמונת פוסטר בפורמט מקודד

טבלת CATEGORIES

טבלת הקטגוריות שומרת את רשימת הקטגוריות האפשריות במערכת  
הטבלה כוללת:

CATEGORIES
CATEGORY_ID מזהה ייחודי לקטגוריה (מפתח ראשי)
NAME_CATEGORY שם הקטגוריה
DESCRIPTION תיאור הקטגוריה
POSTER_IMAGE תמונה ייחודית לקטגוריה בפורמט מקודד

טבלת MOVIE\_CATEGORIES

זוהי טבלת קישור (join table) בין סרטים לקטגוריות.  
היא מאפשרת יצירת קשר מסוג רבים לרבים: סרט יכול להשתייך למספר קטגוריות,  
וקטגוריה יכולה לכלול מספר סרטים. הטבלה כוללת:

MOVIE_CATEGORIES
MOVIE_ID מפתח זר לטבלת MOVIES
CATEGORY_ID מפתח זר לטבלת CATEGORIES



BORROWINGS טבלת

טבלה זו מתעדת את פעולות ההשאלה של סרטים.  
כל שורה מייצגת השאלה אחת ומכילה:

BORROWINGS	
ID_USER	מזהה המשתמש שהשאיל
ID_MOVIE	מזהה הסרט שהושאל
DATE_BORROW	תאריך ההשאלה
DATE_RETURN	תאריך ההחזרה
ID_BORROWING	מזהה ההשאלה (מפתח ראשי)
STATUS	סטטוס ההשאלה

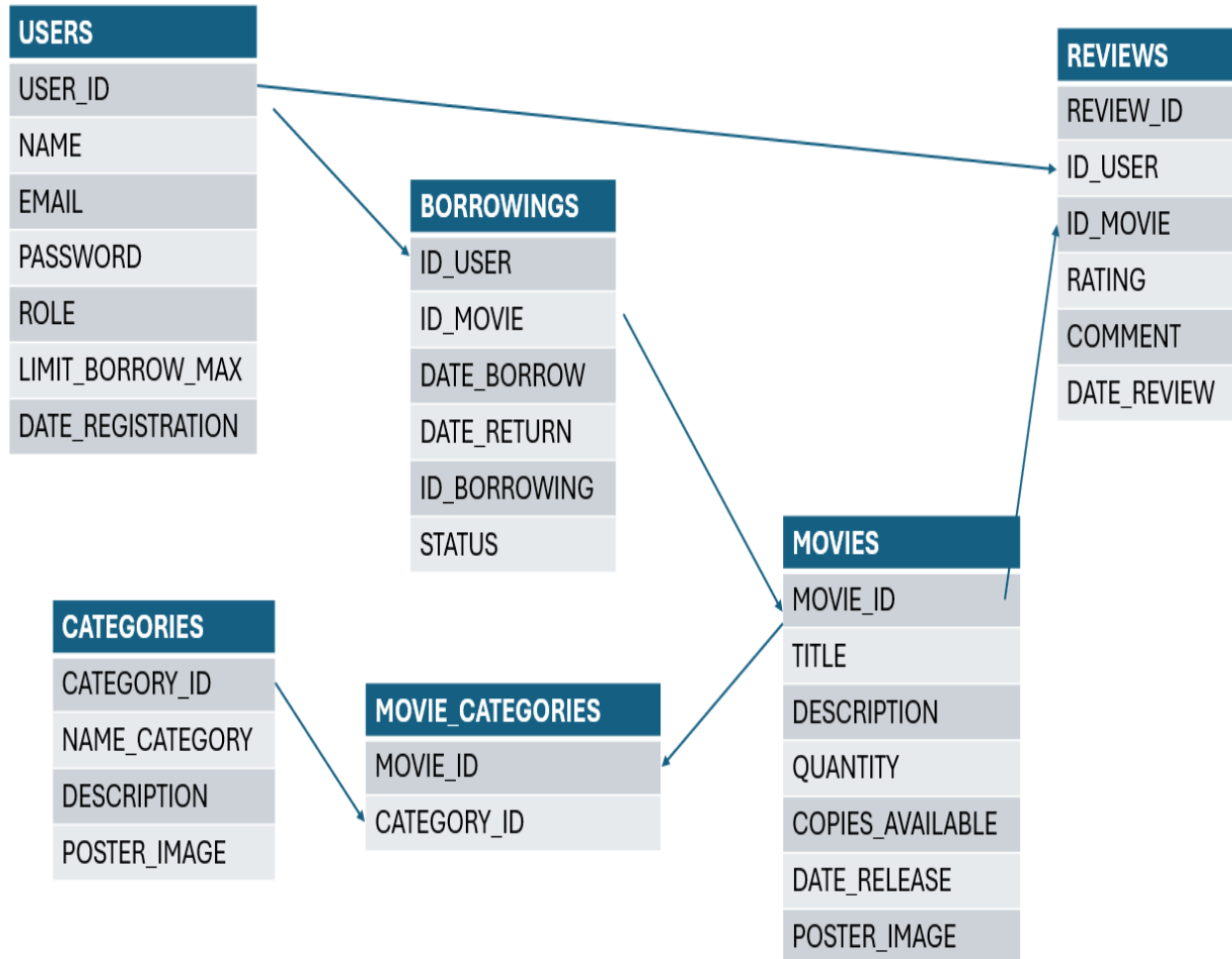
טבלת REVIEWS

בלה זו מכילה את הביקורות שנכתבו על ידי משתמשים על סרטים.  
כל שורה מייצגת ביקורת אחת וכוללת:

REVIEWS
REVIEW_ID מזהה ייחודי לביקורת (מפתח ראשי)
ID_USER מזהה המשתמש שכתב את הביקורת
ID_MOVIE מזהה הסרט המבוקר
RATING דירוג בין 1 ל-5
COMMENT טקסט הביקורת
DATE_REVIEW תאריך כתיבת הביקורת

למערכת יש מגבלה של ביקורת אחת לכל סרט לכל משתמש, והיא נאכפת דרך הלוגיקה באפליקציה.

## תיאור הקשרים במסד הנתונים



## פרק 4 – תיאור המחלקות והקשרים ביניהן

### מבנה כללי

מערכת Almoviland בנויה ממספר חבילות (Packages) הממיינות את המחלקות לפי תפקידן:

- חבילת Bean אחראית על שכבת הלוגיקה והתיווך בין ממשק המשתמש למסד הנתונים.
- חבילת DAO כוללת את מחלקות הגישה למסד הנתונים.
- חבילת Modules מגדירה את הישויות המייצגות את הטבלאות בבסיס הנתונים (Model).
- חבילת Utils מכילה מחלקות עזר.
- חבילת MovieService כוללת את שירות ה-REST
- חבילת config מרכזת את הגדרות המערכת.

הקשרים בין המחלקות נשמרים לפי עקרון MVC, כאשר כל Bean נעזר במחלקת DAO לביצוע פעולות על ישויות ה-Module, ואלו מוצגות לקצה המשתמש בקובצי XHTML דרך JSF.

### מחלקות Bean

מחלקות ה-Bean הן מחלקות מנוהלות (Managed Beans) של JSF. כל מחלקה מייצגת אזור אחר בממשק המשתמש או תהליך עסקי, ומכילה שדות נתונים, מתודות פעולה, והפניות למחלקות DAO. לדוגמה:

– **MovieBean.java** אחראית לניהול הסרטים בצד הלקוח והמנהל. כוללת טעינת רשימות סרטים, סינון לפי קטגוריה, פעולות השאלה, חישוב ממוצע דירוגים, הוספה ועריכה של סרטים ועוד.

– **UserBean.java** מנהלת את המשתמש המחובר, שומרת את המידע עליו (כולל תפקיד), ומספקת מתודות לאימות הרשאות ופעולות נלוות בפרופיל.

– **BorrowBean.java** מרכזת את לוגיקת השאלת הסרטים, החזרות, דיווח על אובדן, ואישורים של מנהלים.

– **ReviewBean.java** מנהלת את הוספת הביקורות, הצגת ביקורות קיימות, מחיקת ביקורת, וחישוב נתונים סטטיסטיים.

– **CategoryBean.java** מטפלת בניהול קטגוריות, כולל יצירה, עריכה, מחיקה וטעינת תמונות.

### מחלקות DAO

מחלקות ה־ DAO (Data Access Object) אחראיות על התקשורת הישירה עם מסד הנתונים. כל DAO תואמת לישות אחת וכוללת שאילתות SQL מוכנות לטיפול בשדות הקשורים לאותה טבלה. המחלקות עושות שימוש ב־ PreparedStatement לשם מניעת SQL Injection.

– **MovieDAO.java** כולל מתודות לשליפת כל הסרטים, חיפוש לפי קטגוריה, עדכון מלאי, הוספה ועריכה של סרטים ועוד.

– **UserDAO.java** מטפל ברישום משתמשים, אימות התחברות, ניהול הרשאות ומעקב אחרי תפקיד המשתמש.

– **BorrowingDAO.java** מנהל השאלות, החזרות, סטטוס אובדן, ואחראי על פעולות עדכון מלאי לפי Borrowing.

– **ReviewDAO.java** מכיל לוגיקה לשליפת ביקורות לפי סרט, הוספה, מחיקה, ומניעת ביקורת כפולה.

– **CategoryDAO.java** מבצע פעולות על קטגוריות ומחזיר נתונים לתצוגה ב־ Bean .

### מחלקות מודל (Modules)

מחלקות ה־ Model הן מחלקות פשוטות שמייצגות ישויות במסד הנתונים. כל מחלקה כוללת שדות מתאימים לטבלה, בנאים, ו־ getter/setter בלבד.

– **User.java** כולל שדות כמו userId, name, email, password, role, joinDate ועוד.

– **Movie.java** מייצג סרט כולל id, title, description, image, quantity ועוד.

– **Category.java** מזהה, שם ותמונה של קטגוריה.

– **Review.java** כולל מזהה ביקורת, מזהה משתמש, מזהה סרט, דירוג, תגובה ותאריך.

– **Borrowing.java** מזהה השאלה, מזהה משתמש, מזהה סרט, תאריך השאלה והחזרה, וסטטוס .

## שירותי REST

במערכת קיים שירות REST בשם **MovieRestService.java**, המחזיר את רשימת הסרטים בפורמט JSON דרך הנתביב:  
<http://localhost:8080/almoviland/api/movies>.  
 השירות מנוהל ע"י המחלקה **RestApplication.java** שמגדירה את הנתביב הבסיסי **/api**.  
 השירות מבוסס על JAX-RS ומאפשר שליפה מהירה של נתונים עבור לקוחות חיצוניים.

## מחלקות עזר ו-Config

- **AppConfig.java** מרכזת הגדרות קבועות של המערכת, כולל כתובת מסד הנתונים, משתמש ברירת מחדל, גבול השאלות ועוד.
- **PageController.java** מנהלת את ניווט הדפים במערכת לפי משתנה **currentPage**.
- **DateUtils.java, MathUtils.java** מחלקות סטטיות לפעולות חישוב ותאריך.
- **PasswordMatchValidator.java** מחלקת ולידציה מותאמת ל-JSF לבדיקת התאמה בין סיסמה לאימות סיסמה בעת רישום.

## פרק 5 – קבצי XHTML והקשר למחלקות

`home.xhtml` - מציג את עמוד הבית עם רשימות של סרטים פופולריים וסרטים לפי קטגוריה. משתמש ב- `movieBean` וב- `categoryBean` כדי להציג סרטים ולנווט בין עמודים באמצעות AJAX.

`movieDetails.xhtml` - מציג מידע מלא על סרט שנבחר, כולל תמונה, תיאור, תאריך יציאה, כמות זמינה, דירוג ממוצע, וכפתורים לפעולה (השאלה, הוספת ביקורת, צפייה בביקורת). מבוסס על : `movieBean`, `reviewBean` `borrowBean`.

`viewReviews.xhtml` ו `addReview.xhtml` - מאפשרים למשתמשים רשומים להוסיף ביקורת על סרט או לצפות בביקורות קיימות. ב `viewReviews.xhtml` קיימת הרשאת מחיקה לבעל הביקורת ולמנהל. שניהם פועלים דרך `reviewBean`.

`profile.xhtml` - מאפשר למשתמש לצפות בהשאלות הפעילות שלו, להחזיר סרטים, לדווח על אובדן, ולצפות בהיסטוריה. מחובר ל `movieBean`, `borrowBean`, `userBean`.

`login.xhtml` ו `register.xhtml` - עמודים שמנהלים תהליך ההרשמה וההתחברות של המשתמש, כולל ולידציה של סיסמה, אימות אשראי, הגבלת אורך סיסמה, ודפוס אימייל. מבוססים על `userBean`.

`adminDashboard.xhtml` - מכיל קישורים עיקריים עבור מנהל: ניהול סרטים, קטגוריות, משתמשים, ובקשות לאובדן. לא כולל לוגיקה עסקית אלא רק ניווט.

`editMovie.xhtml` ו `moviesManagement.xhtml` - מאפשרים למנהל להוסיף סרטים חדשים, לערוך פרטי סרט קיים, ולבצע פעולות על המלאי. משתמשים ב `movieBean`.

`editCategory.xhtml` ו `categoriesManagement.xhtml` - עמודים המאפשרים הוספה, עריכה ומחיקה של קטגוריות כולל תמונה, דרך `categoryBean`.

`usersManagement.xhtml` ו `userBorrowingHistory.xhtml` - עמודים לניהול המשתמשים – צפייה במידע, שינוי תפקידים, מחיקה, וצפייה בהיסטוריית ההשאלות. מבוססים על `userBean`.

`reportLoss.xhtml` ו `adminLossRequests.xhtml` - משתמש רגיל מדווח על אובדן סרט עם פרטי אשראי, ומנהל מאשר את האובדן דרך טבלת אישורים. מחובר ל `borrowBean`.

`search.xhtml` - ממשק חיפוש דינמי עם תצוגת תוצאות מיידית לפי טקסט שהוזן. נתמך על ידי `movieBean` וכולל AJAX עם `keyup`.

### קומפוננטות משותפות והרכבת הדפים

מערכת Almoviland עושה שימוש בקובץ index.xhtml כשלד הראשי של הממשק. בתוך קובץ זה נטענים עמודי התוכן באופן דינמי לפי ערך משתנה currentPage שב- PageController, תוך שימוש ברכיב <ui:include> ו <h:panelGroup> .

בראש כל עמוד נטען באופן קבוע קובץ header.xhtml, שמכיל את תפריט הניווט הראשי. תפריט זה מותאם להרשאות המשתמש:

- אם המשתמש אינו מחובר – מוצגים קישורים ל Register ו Login .
- אם המשתמש מחובר מוצגים שמו כפתור Logout ו Profile .
- אם המשתמש הוא Admin מוצג גם קישור ל Admin Dashboard .

בנוסף לכך, הודעות מערכת (שגיאות, אישור, מידע) מוצגות בכל עמוד על ידי רכיב <h:messages> אשר נטוע באופן ישיר בכל קובץ שבו יש צורך בהצגת תגובה.

כל הדפים שומרים על סגנון אחיד בעזרת קובץ CSS מרכזי styles.css וקובץ JavaScript scripts.js . המוטמעים דרך <h:outputStylesheet> ו <h:outputScript> .

כך נוצר ממשק אחיד, רספונסיבי, המופעל כולו מתוך שלד אחד – ומוצגים בו תכנים משתנים באופן מודולרי.

### סיכום

המבנה של Almoviland משתמש במערכת טעינה דינמית של רכיבים לפי pageController, בכך מאפשר שליטה קלה על ניווט, הצגת מסכים לפי הרשאות, והפרדה בין תפקידים. כל XHTML מקושר ישירות למחלקת Bean אחת לפחות, וכל פעולה בממשק מעודכנת בזמן אמת בצד שרת דרך AJAX .



## פרק 6 – זרימת מידע בין שכבות

מערכת Almovidland פועלת לפי עקרונות MVC (Model-View-Controller) כאשר הנתונים נעים בין שכבת התצוגה (XHTML), שכבת הלוגיקה (Beans) ושכבת הגישה לנתונים (DAO).

זרימת המידע מבוססת על שימוש ב JSF, כולל תגיות binding לשדות ב Managed Beans.

ועבודה עם AJAX לעדכון דינמי של תוכן בדף ללא רענון מלא.

כאשר משתמש מבצע פעולה בממשק, לדוגמה: לוחץ על כפתור "השאל סרט", הפעולה מועברת דרך כפתור `<h:commandButton>` או `<f:ajax>`. כפתור זה מפעיל מתודה מתוך Bean מתאים.

המתודה ב- Bean אחראית לבדוק תנאים (כמו מגבלת השאלות או זמינות העותקים) ולאחר מכן קוראת למחלקת DAO אשר מבצעת את העדכון במסד הנתונים.

ה-DAO מחזיר את תוצאת הביצוע וה-Bean בהתאם מפעיל לוגיקה עסקית משלימה: הצגת הודעה למשתמש, ריענון נתונים, או שינוי דף. אם יש צורך, ה-Bean יעדכן גם שדות אחרים (כמו כמות זמינה בסרט או היסטוריית השאלות בפרופיל המשתמש).

במקביל, הנתונים שמוצגים על המסך (כמו רשימת הסרטים או פרטי המשתמש) נקראים מתוך ה-Bean ומועברים לתצוגה באמצעות EL.

שדות בטפסים כמו comment, rating, email וכו' מקושרים לשדות ב-Bean, כך שהשינויים שמבצע המשתמש מועברים אוטומטית לשרת דרך המנגנון של JSF.

מערכת ההודעות `<h:messages>` מציגה תגובות מהשרת (למשל: הצלחה, שגיאה, התראה), בהתאם לפעולות שבוצעו.

כמו כן, נעשה שימוש ב `<f:ajax>` להצגת תוצאות חיפוש בזמן אמת, לעדכון חלקים בדף (כמו רשימת סרטים לפי קטגוריה או תוצאות חיפוש), ובכך נשמרת חוויית משתמש חלקה ואינטראקטיבית.

זרימה זו חוזרת על עצמה בכל פעולה במערכת – רישום, התחברות, השאלה, החזרה, דירוג, ניהול ועוד – וכל רכיב במערכת מממש את חלקו לפי תפקידו: View מציג Bean מתוך DAO ניגש למסד.

## פרק 7 – היבטים מערכתיים

### אבטחת מידע

המערכת כוללת מספר מנגנוני אבטחה בסיסיים לשמירה על פרטיות המשתמשים ושלמות המידע. סיסמאות המשתמשים נשמרות במסד הנתונים לאחר הצפנה באמצעות אלגוריתם Hash מסוג SHA-256, כך שלא ניתן לקרוא אותן בצורה גלויה. כל הפעולות מול מסד הנתונים מבוצעות באמצעות PreparedStatement דבר שמונע התקפות מסוג SQL Injection. כמו כן, למשתמשים יש גישה רק לאזורי המערכת המותאמים לתפקידם – משתמש רגיל לא יכול להיכנס לעמודי ניהול, והלוגיקה ב JSF שומרת על כך גם בצד התצוגה וגם בצד השרת.

### בקרת הרשאות ותפקידי משתמש

במערכת קיימים שני תפקידי משתמש: ADMIN ו USER. ההרשאה נקבעת בעת ההרשמה או על ידי שינוי תפקיד דרך ממשק הניהול. ההרשאה משפיעה על התפריטים המוצגים, הדפים הזמינים, והפעולות שניתן לבצע (כמו מחיקת משתמשים, ניהול מלאי, אישור אובדן ועוד). כל עמוד XHTML כולל תנאי rendered שמונע הצגת רכיבים למשתמשים שאינם בעלי ההרשאה המתאימה, ובצד ה Bean קיימת לוגיקה שמבצעת בדיקות חוזרות גם על הפעולות עצמן.

### ולידציה ובקרת קלט

המערכת מבצעת ולידציה (אימות תקינות קלט) הן בצד הלקוח והן בצד השרת. לדוגמה: בדיקת אימייל, התאמת סיסמא לאימות סיסמא, תקינות של כרטיס אשראי (אורך תוקף ו CVC). בדיקות אלו נעשות באמצעות תבניות pattern, הודעות שגיאה מותאמות, ולוגיקת אימות פנימית ב Beans. בשדות כמו comment ו rating נעשית ולידציה למניעת הגשה לא תקינה של ביקורות.

### מניעת התנגשויות בשאלת סרטים

כאשר משתמש מנסה להשאיל סרט, המערכת מפעילה שאילתת SQL שמורידה עותק מהכמות הכוללת רק אם קיימים עותקים זמינים גדולה מ 0. העדכון מתבצע בפעולה אחת – כך שגם אם שני משתמשים לוחצים על "השאל" במקביל, רק אחד יצליח. המערכת מונעת בכך השאלת יתר ושומרת על עקביות מלאה במלאי.