



המכללה המדעית טכנולוגית

אורט ביאליק



המכון הממשלתי להכשרה
בטכנולוגיה ומדע

מגמת הנדסת תוכנה

מסלול: הנדסאי תוכנה

פרויקט גמר: *HR-Pulse*



אפליקציה לניהול מערכת משאבי אנוש בחברה קטנה עד בינונית.

מגישי: סרגיי מזורוב, ת.ז: 321320384

עובד אלה מסרי, ת.ז: 302979687

מנחים: מוטי פניצ'ישווילי ואודי מלכה.

רכז מגמה: מוטי פניצ'ישווילי.

תאריך הגשה: 08.02.2024

1.	3	רקע על הפרויקט
2.	11	דיagramות של המערכת
3.	14	התיחסות לנושאי אבטחת מידע
4.	16	תבנית עבודה ושלבים לימוש הפרויקט –
5.	18	תכנון הבדיקות, מודולים ב.יחדה
6.	20	דף כניסה ראשי של המשתמש
7.	22	דף ניהול עובדים
8.	28	דף שינוי/עדכון משמרת
9.	35	דף משוב לעובד
10.	37	דף ניהול מחלקות
11.	41	דף הפקט דוחות
12.	46	פרטי העובדים בסיס נתונים
13.	48	כלל קבצי התוכנה
14.	Module info	137
15.	JRXMLs view	137
16.	POM dependencies file	153
17.	View XMLs	155
		פרונט-אנד

1. שם הפרויקט – HR-Pulse

2. רקע:

2.1. תיאור ורקע כללי:

מערכת מידע לניהול משאבי אנוש (HRMIS) מיועד לחברות קטנות עד בינוניות, התוכנה תנהל את המבנה של החברה:

1. מבחינת מחלקות – הוספה/הסרת מחלקות, צירוף עובדים למחלקות והסרתם וכו'.
2. מבחינת עובדים – הוספה/הסרת עובדים, תפקידם בארגון/חברה, שכר, פרטיים אישיים וכו'.
3. ניהול נוכחות עובדים - מבחינת שעות עבודתם, הפסקה, מחיקת נתונים והוספתם וכו'.

יהיה ניתן להפיק מהתוכנה דוחות חיוניים כגון – דוח שעות חדש של עובד ודו"ח עובדים כללי.

התוכנה גם מאפשרת קליטת נתונים נוכחות (שעות עבודה מתוך קובץ CSV, עם שמירה למסד נתוניםväגיבוי).

2.2. מטרות המערכת:

לאפשר לאנשי החברה לעבוד דרך תוכנה אחת התousel את רוב המידע הנחוץ לניהול משאבי האנוש, ובכך תצמצם עלויות על כוח אדם נוספת. התוכנה מאפשרת לאנשי צוות בדרגות השונות בחברה להיכנס רק לאופציות המתאימות לה לפי דרגתו.

3. סקירת מצב קיימם בשוק, אילו בעיות קיימות?

קיימות מספר תוכנות לניהול חברה ועובדים בשוק, Oracle HCM, SAP HR, UltiPro ורבות אחרות לניהול חברות קטנות עד גדולות. לעיתים הן יקרות מאוד ודורשות תחזוקה של יומ-יומ, תוספות של פיצרים והרחבות שעולות עוד הון-תועפות. לכן הבעייהינה אינטגרציה ועלויות גבוהות אשר חברות קטנות עד בינוניות לא תמיד מסוגלות לאפשר לעצמן לרכוש. מרכיבות התוכנהינה עד בעיה ידוע, היא יכולה להיות מסורבלת וקשה לשימוש, כך שהיא משתמש ילווה באימוץ מיותר וירידה בפראודקטיביות בין אם בזמן הקוצר או הארוך.

4. מה הפרויקט אמר לחידש או לשפר?

הפרויקט הנ"ל יביא אליו פתרונות אוטומציה(לאחר הэнזנות התחלתיות ידניות) שיוציאו דוח שעות העובדים והעבדים בחברה ככאשהມידע יהי נגיש וקל לגורם הרלוונטי. כמו יופחת השימוש הדני ועליה מניע של טעויות ידניות אפשריות. בנוסף, הפרויקט יאפשר לארגונים לקבל תובנות עמוקות יותר על נתונים כוח האדם שלהם ולקבל החלטות מושכלות יותר בפשטות וקלות ללא סרבולים ושמירת הנתונים בו בזמן במסד הנתונים וקובץ CSV במקרה של שעות הנוכחות.

5. דרישות מערכת ופונקציונליות

5.1. דרישות מערכת:

סביבת הטעמה ושימוש. שרידות, ביצועים ותאמודות עם עומסים.

- א. מחשב עם מערכת הפעלה של וינדואס.
- ב. מסד נתונים MySQL.

ג. טעינת נתונים שעות מtower קובץ (מסוג CSV) של שעון נוכחות (יסוכם עם החברה).

ד. תוכנת JasperReports להפקת דוחות.

5.2. דרישות פונקציונליות:

רשימת דרישות המשמש מהמערכת, מהן הפעולות בהן נדרש המערכת לתמוך.

א. ניהול מאגר משתמשים וכינסה לתוכנה - משתמשי מערכת.

ב. ניהול כרטיסים עובד.

ג. ניהול מחולקות בחברה וחלוקת תפקידים לעובדים.

ד. ניהול שעות עבודה של העובדים.

ה. הפקת דוחות – נניח ליצירת דוח שעות חודשי לעובד, שאותו ניתן להעיבר לרוחת חשבון.

6. בעיות צפויות במהלך הפיתוח ופתרונות (תפעוליות, טכנולוגיות, עומס ועוד):

6.1. תיאור הבעיה- הלו כפועל יוצא של דרישות המשמש מהתוכנה.

א. בעיות טינה לבסיס הנתונים ע"י הכנסת נתונים לא נכונים/תואמים.

ב. שרת בסיס נתונים אינו זמין בעת שמירת הנתונים מהטבלה של שעות עבודה.

ג. Maven לאמצא אותו ספריות מסוימות.

ד. במהלך קריסת שרת/אינטרנט וכו' – לא יהיה ניתן להתחבר לאפליקציה בשל אי זיהוי תפקיד.

6.2. פתרונות אפשריים. (נא ציין פתרונות אפשריים וחולופות ארכיטקטוניות)

א. בעת שימוש יזין פרטימ לאריך נתונים בעמודות - יקבל הטרעה לתקן ובאייה פורמט עלייו לתקן.

ב. יהיה ניתן לשמר במקורה הנ"ל בקובץ CSV – וכאשר השרת יחזור לפועל – יהיה ניתן לשמר מהקובץ CSV למסד הנתונים ככה ששניהם תמיד יהיו מסונכרנים.

ג. הטמעת הספריות ידנית מהאתר <https://mvnrepository.com> לאפליקציה דרך סביבת העבודה IntelliJ IDEA.

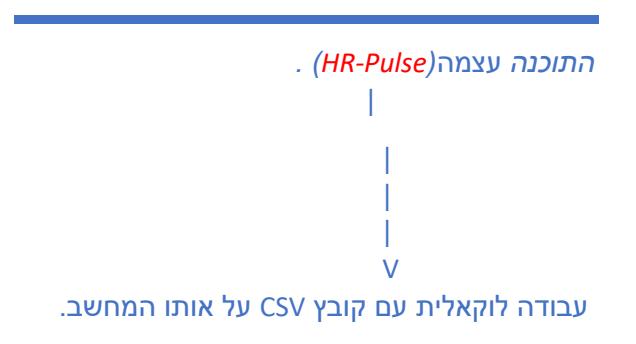
ד. הכנסנו משתמשים HARD CODED לכל תפקיד והגורמים הרלוונטיים תהיה נא גישה אליהם ויכולו להיכנס דרך האפליקציה ללא צורך במשתמש שלהם דרך בסיס הנתונים.

7. פתרון טכנולוגי נבחר:

7.1. טופולוגיה הפתרון - כלומר : פרישת המערכת , היכן יתבצע יישום המערכת (deployment), מרכבי הפרישה. הניל ברמת מערכת (לדוויי פרויקט פיתוח אתר אינטרנט : המערכת מורכבת משרת, ממשק משתמש מצד הלוקוח, DB's, טווח תקשורת-אינטרנט, המערכת תיושם בראשת האינטרנט , יש להציג את דיאגרמת המערכת וכו')

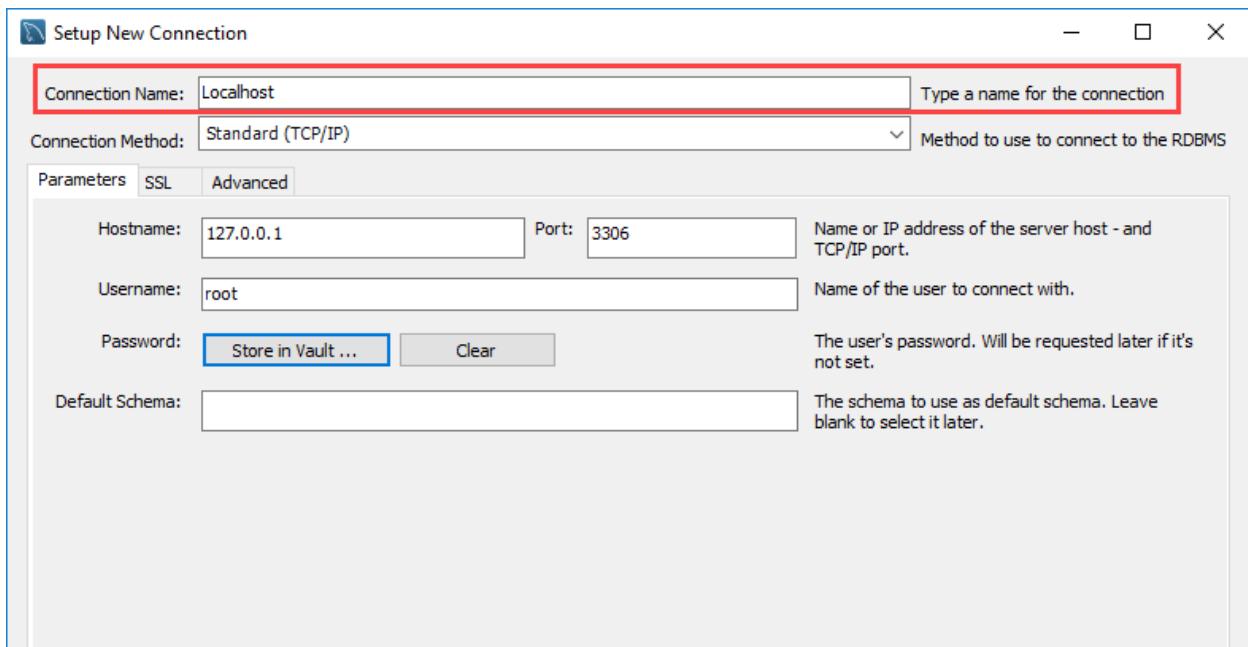
אפשרות ראשונה:

המערכת תעבור לקהל, ז"א עבודה עם שעות פעילות העובד יכולות להתבצע בעזרת קובץ CSV בלבד. ככה לא תהיה תלות בשינוי שעות עבודה אם נצטרך בדחיפות.



אפשרות שנייה:

בסיס הנתונים ישב על העמדת הנוכחית(או אחרת) בראשת המקומית LAN והميدע יעבור בקו התקשרות בסיס הנתונים MySQL.



7.2. טכנולוגיות בשימוש.(איזה ומדוע בכמה מילימס) :

- IDEA – הינה סביבת פיתוח אשר מצאנו בו ממשק ידידותי ומתקדם גם עם הכלים שהוא מציעה וגם עם מערכת הסיוו שלוי בכתיבת קוד.
- JavaFx זו פלטפורמה רבת עצמה ורב-תכליתית לייצור אפליקציות שלחניות ומובייל חזות פלטפורמות עם ממשק משתמש עשירים **יכלולות גרפיות מתקדמיות**.
- SceneBuilder – כלי המשמש **בשילוב** עם JavaFx. הוא מספק ממשק חזותי - "גרור-ו-שחרר" לעיצוב ופריסה של ממשקים. הכללי נבחר כליל לחיסכון בזמן העיצוב.
- OpenCSV- ספרייה אשר מאפשרת לנו לכתוב נתונים (לתרגם) לקובץ CSV, הספרייה הזאת תביא לנו את האפשרות לרשום מערך/רשימה של אובייקטים או מערך של מחוזות בצורה ישרה.
- Hibernate – הינה אינטראקציה עם Database בעזרת עובדה עם אובייקטים של Java במקום לעבוד עם שפת SQL .
- JasperReports – הינה ספרייה המאפשרת הפקה של דוחות ובמסמכים שונים באפליקציה, לאחר הפתק הדוחות (כגון ביצועי עובדים, תקציבי מחלקה וכו') יהיה ניתן לשתף אותם עם בעלי עניין/מוצג רלוונטיים.
- JavaMail – הינה API Java המאפשר לשלווה ולקלות הודעות דואר אלקטרוני מתוך יישום Java. זה יכול לשמש לדוגמא – שליחת דוא"ל למנהלים על הוספה חדשה, שינוי סטטוס עבודה (גם חוליה/חזר מחלקה/בריא), או לעדכן לגבי עובד שהוקצה למחלקה מסוימת.
- MySQL – מסד נתונים של אורקל.

7.3. שפות הפיתוח: (איזה שפות ומדוע בכמה מילימס?)

- בחירה שפת הפיתוח Java,
התוכנה נכתבת בשילוב טכנולוגיות שונות, עדכניות וROLONTIOTOT לבניית תוכנה למחשב שלחני אשר צורכת בסיס נתונים MySQL לוקאלי או ברשת Lokalit. במידה והתוכנה תעבור מכמה תחנות (גנich מחשבים במחלקות שונות), חיבת להוות גישה לכל התחנות לבסיס הנתונים LAN).

7.4. תיאור הארכיטקטורה הנבחרת- הסבר בכמה מילימס מדויק :

- הארQUITECTURA שנבחרה הינה **שרט-לקוח Client-Server Architecture**.
מספר משתמשים בדרגים שונים צריכים לגשת לאפליקציה והם עשויים להיות במחלקות שונות, האפליקציה תספק גישה מאובטחת לתפקיד הרלוונטי.
בארכיטקטורה זו, האפליקציה תחולק לשני חלקים עיקריים: **צד לקוח**, שהוא ממשק איתנו משתמשים (מנחים, מזכירה וכו') מקימים אנטראקציה. **צד השירות**, שמנהל את הנתונים והלוגיקה העסקית של האפליקציה - יהיה אחראי על אחסון וניהול הנתונים הקשורים לעבודים, מחלקות, שעות נוכחות וכו'.
- צד הלקוח** יספק את המשק למשתמשים כדי לגשת ולשנות את המידע שלעיל.

7.5. חלוקה לתוכניות ומודולים :

מודולים בצד הלוקה:

1. **מודול ממשק משתמש (עט)**: מטפל באינטראקציה של המשתמש עם השירות ומציג מידע רלוונטי למשתמש.

2. **מודול גישה לנוטונים**: מנהל אחסון ואחיזור נתונים מקומיים, כגון קריאה וכתיבת לקובץ ה-.CSV

תוכנית בצד הלוקה:

תוכנית ראשית: מתחילה את המודולים השונים בצד הלוקה, מטפלת בקלט המשתמש ומתקשרת עם התוכנית בצד השירות.

מודולים בצד השירות:

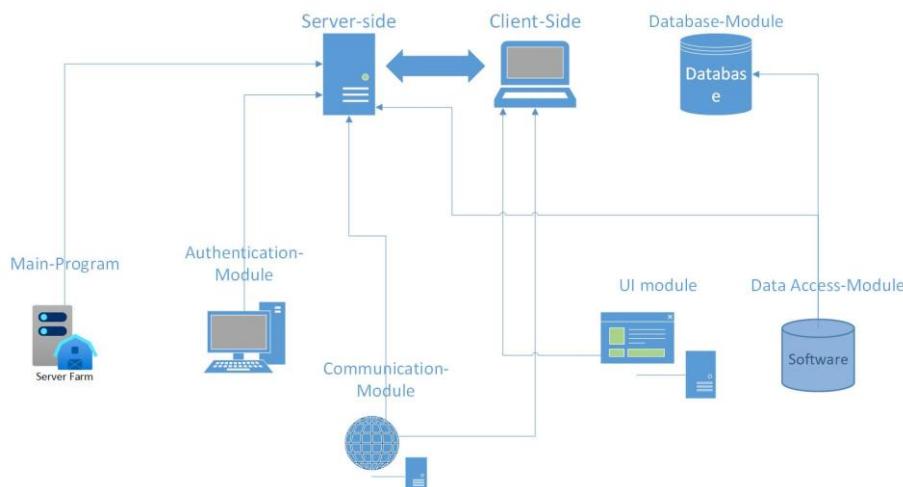
1. **מודול אימות**: מטפל באימות והרשות של משתמשים, ובतוךו שרך משתמשים מורשים יכולים לגשת לאפליקציה.

2. **מודול מסד נתונים**: מנהל את מסד הנתונים של האפליקציה, אחסון ואחיזור מידע על עובדים, מחלקות, משמרות וכו'.

3. **מודול תקשורת**: מאפשר תקשורת בין אנשי הצוות ע"י שליחת אימייל דרך JavaMail.

תוכנית בצד השירות:

תוכנית ראשית: מרכזת את המודולים השונים בצד השירות, מטפלת בבקשתות כניסה מהתוכנית בצד הלוקה ושולחת תשובות בחזרה ללוקה.



- התוכנית בצד הלוקה מתקשרת עם התוכנית בצד השירות ולהיפך, כדי לשולח ולקבל נתונים ובקשות/תשובות.
- התוכנית הראשית בצד הלוקה מתקשרת עם התוכנית בצד השירות כדי לתחם את המודולים בצד הלוקה ולטפל בבקשתות ותשובות המשתמשים.

- מודול האימוט מתקשר עם התוכנית בצד השרת כדי לאמת ולהעניק הרשאה למשתמשים ולהבטיח שרק משתמשים מורשים יכולים לגשת לאפליקציה.
- מודול התקשרות מתקשר הן עם התוכנית בצד השרת והן עם צד הלוקה כדי לאפשר תקשורת בין מנהלי מערכת.
- מודול משק המשתמש מתקשר עם התוכנית בצד הלוקה כדי לטפל באינטראקציות של המשתמשים ולהציג מידע למשתמש.
- מודול הגישה לנוחים מתקשר גם עם מודול מסד הנתונים וגם עם התוכנית בצד השרת כדי לנוהל אחסון ואחזור נתונים מקומיים, וגם לתקשר עם התוכנית בצד השרת כדי לשולח ולקבל נתונים.

7.6. סביבת השרת (מקומי, וירטואלי, ענן, שירות אירוח) :

לבחירה הלוקום:

- כאופצייה ראשונה: ● ללא סביבת שרת כלל, הכל נשמר ועובד על המחשב עצמו.
- כאופצייה שנייה: ● שרת תקשורת לוקאל.

7.7. משק המשתמש/локוח – GUI :

תיאור	רכיב	מודול
אפשר למשתמש להיכנס לאפליקציה עם שם המשתמש והסיסמה שלהם.	מסמך כניסה/התחברות	(ו)טפסק משתמש ()
מספק תצוגה ברמה גבוהה של אופציות האפליקציה, כולל גישה מהירה לתוכנות בשימוש תקוף.	מסמך ראשי(מסמך מחוונים)	
אפשר למנהל מחלקות להוסיף, להסיר ולשלוט בנוחונים על עובדים במחלקה שלהם.	ניהול מחלקה	
אפשר להציג ולנהל נתונים על עובדים.	ניהול עובדים	
אפשר למשתמשים ליצור ולהפיק דוחות על סמך הצריכים שלהם(דו"ח עובדים/מנהלים).	הפקת דוחות	

אפשר למשתמשים לייבא ולייצא נתונים מקובץ CSV במחשב המקומי שלהם.	ייבוא/ייצוא של CSV	גישה לנוחנים
אפשר למשתמשים להתחבר למסד נתונים של MySQL ברשות מקומית לאחסון וஅחזר נתונים.	חיבור מסד נתונים של SQL	

7.8. ממשקים למערכות אחרות / API :

לא קיימים למערכת ממשקי API אך ניתן לבנות כלו בעתיד על פי הצורך. במידה ונרצה לבנות אפליקציות
móvel שמתממשקות עם התוכנה.

7.9. שימוש בחבילות תוכנה :

- IntelliJ IDEA

- JavaFx

- SceneBuilder

- OpenCSV

- Hibernate

- JasperReports

- JavaMail

- MySQL

8. שימוש במבני נתונים וארגוון קבצים

8.1. נא פרט את מבני הנתונים :

(האופציות יהיו ניתנות לשינוי במהלך בניית הפרויקט לאופטימליות מתאימה)

- רשימות הקשורות(Linked-List) ומערכות לאחסן מידע על עובדים, מחלקות ומנהלים.
- מפות(Maps) כדי לאחזר מידע במהירות על סמך מפתחות כגון ת.ז, או כל מפתח שייתן כניסה לשורה הרלוונטייה בטבלה.
- מערכאים(Arrays) לאחסן ומניפולציה של עובדים, מידע ופרטים רלוונטיים אחרים.

באשר לארגו הקבצים, ניתן יהיה לגשת אליהם בשיטות הבאות:

- אחסון נתונים במסד נתונים MySQL, אותו ניתן לארgo בטבלאות ולהפיק שאלות באמצעות עובודה עם Hibernate.
- אחסון נתונים משמרת בקובץ CSV בפורמט מובנה וモתאם/מסוכם עם מכשיר הדפסת שעות. תאריך, שעת כניסה, יציאה, סה"כ שעות עובודה וכו'.

8.2. נא פרט את שיטת האחסון (מאגר, קבצים ובאייה טכנולוגיה):

עbor אפשרות 1 (באמצעות LAN מקומי ומסד נתונים של MySQL):
נשותמש ב-Hibernate כדי לנהל את הנתונים במסד הנתונים. מסד הנתונים ישמש כמאגר הנתונים, מאורגן בטבלאות עם שורות המיצגות רשומות שונות ועמודות המיצגות את התכונות של כל רשומה – מסד הנתונים היה מקור מרכזי לשמירת כל נתונים העובדים, מחלקות, משמרות וכו'.

עbor אפשרות 2 (באמצעות קובץ CSV באוטו מחשב):
נשותמש בספרייה OpenCSV כדי לקרוא ולכתוב נתונים לקובץ CSV. קובץ CSV ישמש כמאגר עבור הנתונים, מאורגן בפורמט טבלה עם שורות המיצגות עמודות שונות לאחסון ומיניפולציה של משמרות העובד.

- מבחרית ארגון הקבצים, ניתן יהיה לאחסן את קובץ ה-CSV בספריה ספציפית במחשב המקומי, בעוד שבסיס הנתונים יהיה על שרת LocalHost

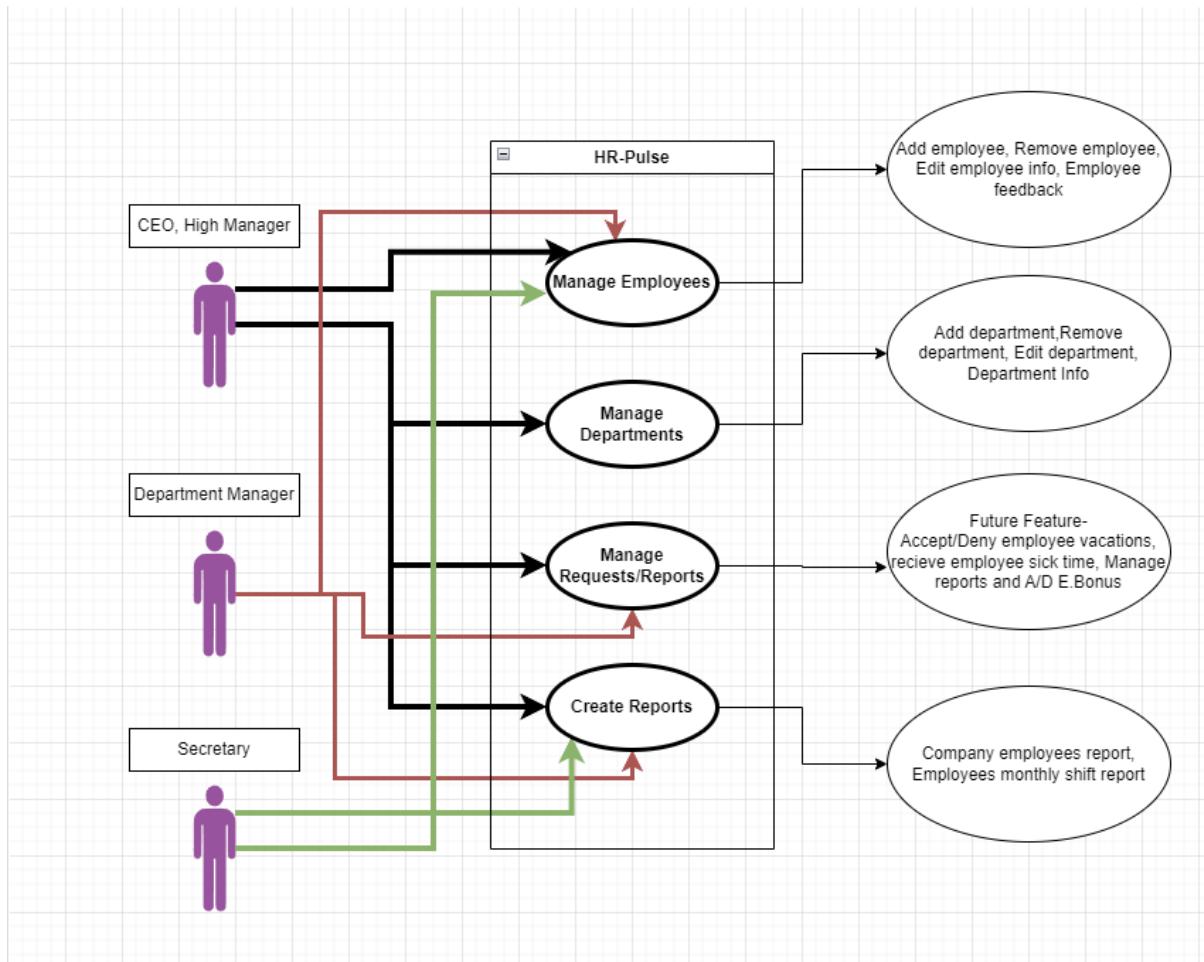
8.3. נא ציין מנוגני התאוששות מנפילה וקריסה תמייה בטראנזקציות:

פונקציונליות גיבוי ו恢復:

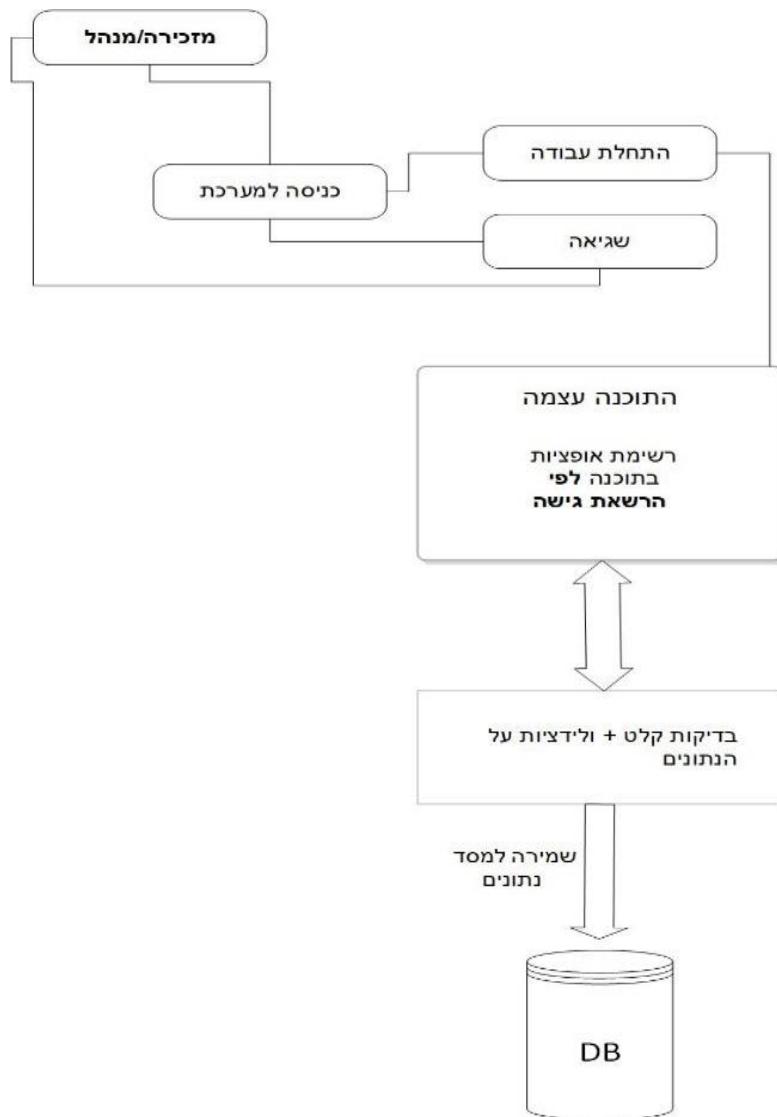
- בעבור קובץ ה-CSV, השמירה תהיה ע"י לחיצת כפתור "שמור" ולאחר מכן המידע יישמר גם בקובץ CSV וגם במסד הנתונים – היה וקובץ ה-CSV הלא לאיבוד/نمתק – יהיה ניתן בפתור "נתונים אחרים" לשחזר את כל הנוגע למשמרות העובד ולשמור מחדש בקובץ ה-CSV.
- בעבור אפשרות MySQL, ניתן יהיה להשתמש בתכונות גיבוי ו恢復 של שרת MySQL כדי ליצור גיבויים תקופתיים של מסד הנתונים.

9. תרשימי מערכת מרכזים

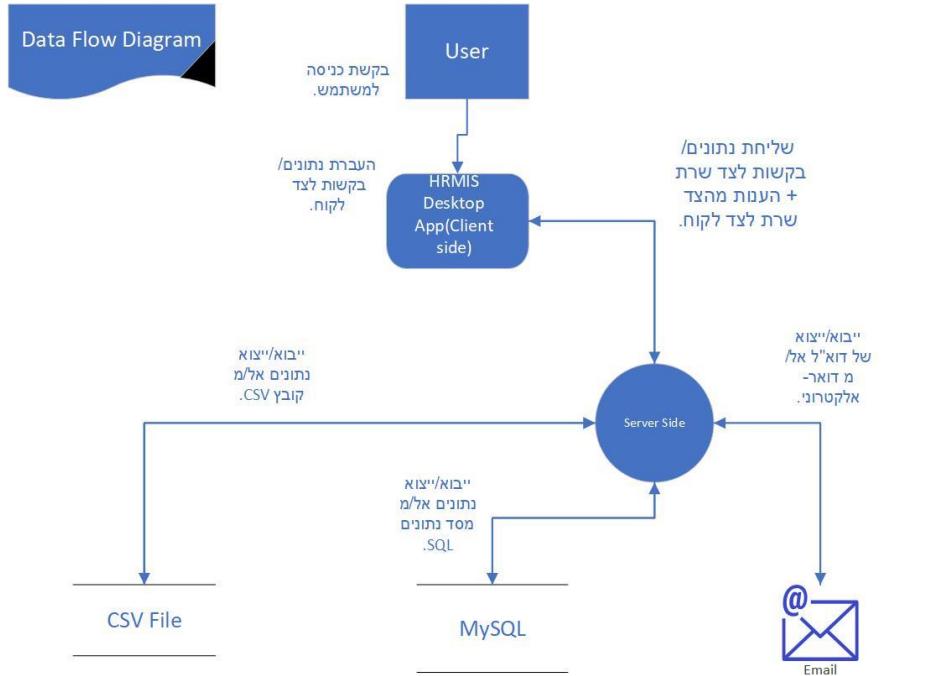
Use Case .9.1



**רץ קריאות פונקציות מרכזיות בלוגיקה העסקית Sequence Diagram .9.2
המרכזיות של הפרויקט**



-Data Flow .9.3



האפליקציה עצמה(צד לקוח) :HRMIS App(HR-Pulse)

שם: אפליקציה שולחן העבודה של HRMIS

תיאור: האפליקציה העיקרית שאיתה משתמשים מקימיים אינטראקטיבית ניהול מחלקות, עובדים, ראשי מחלקות, שכר עובדים, משירות, בינויים, משוב והפקת דוחות בקיצור לצד שרת.

משתמש : User

שם: משתמש

תיאור: האדם המקיים אינטראקטיבית עם התalic האפליקציה כגון ניהול משאבי אנוש, ניהול מחלקה, מצירה או בעלים וכו'(המערכת תזהה את המשתמש לפני הרשות הגישה ותיתן גישה לאופציות הרלוונטיות בדרגת).

קובץ CSV:

שם: קובץ CSV נתונים לאילם.

תיאור: מאגר הנתונים שבו המשתמש יכול ליבא או ליצא מידע עבור.

אפליקציה קידמית שרת :Server App

שם: אפליקציית שרת

תיאור: התהילה האחראי על ניהול נתונים חברה בצד השרת של האפליקציה – מהו שיקר בין הנתונים למשתמש וכו'.

מוד נתונים SQL:

שם: MySQL

תיאור: מאגר הנתונים שבו מאוחסנים נתונים החברה בצד השרת של אפליקצייה.

amil :Email

שם: אמייל

תיאור: כלים המשמשים לתקשורת בין משתמשים ולשיתוף דוחות שנוצרו.

10. תיאור המרכיב האלגוריתמי – חישובי

נוסף מידע כגון שכר העובדים (chodshi/yomi), איזה תאריכים בא לעבודה, ונפיק דוח למשתמש הרלוונטי. הנתונים יישמרו בקובץ CSV וברשות הנתונים MySQL. המערכת כפי שצין – תפיק דוח לגורמים הבכירים יותר באמצעות המזקירה/מנהל כדי לקבל רקוּן כלִי על העובד/ים או מה שנוגע למשמרות שלהם שלאחר מכן, יהיה ניתן לחתם את הדוח לרשות חשבון אשר בקהלות תוכל להציג את סה"כ שעות העבודה של עובד מסוים לפי חדש ולהפקיד את שכוֹר.

11. תיאור/התיחסות לנושאי אבטחת מידע :

נא לציין אזורים הדורשים אבטחה, כגון : שרת, בקרת כניסה לאתר, חשבונות משתמשים, מאגרי מידע וכיוצא ניתן מענה :

1. אימות משתמש :

איומים : כניסה לא מורשית למערכת, גניבת מידע, שיבוש נתונים.

תגובות : הגבל ניסיונות כניסה והקצת זמן בין עוד סריה של כניסה.

2. בקרת כניסה לנתונים :

איומים : כניסה לא מורשית לנתונים רגילים, דליפת נתונים ושיבוש נתונים.

תגובות : הטמעת בקרת כניסה מבוססת תפקדים, הגבלת כניסה לנתונים למשתמשים מורושים בלבד.

3. גיבוי ו恢復 נתונים :

איומים : אובדן נתונים עקב כשל בחומרה, טעות ידנית/אנושית.

תגובות : גיבוי נתונים באופן קבוע למקום מאובטח, יישם תוכניות התאוששות ע"י גיבוי בשרת MySQL עצמו.

4. (חשיבות) בטיחון פיזי :

איומים : גניבה, נזק פיזי לשרת או למחשב וגישה לא מורשית.

תגובות : הטמעת אמצעי אבטחה פיזיים כגון מנעלים, מצלמות אבטחה ומערכות בקרת כניסה.(החלק הנ"ל הינו על הלוקוט).

12. משאבי הנדרשים לפרויקט :

12.1. מספר שעות המוקדש לפרויקט, חלוקת עבודה בין חברי הצוות :

חלוקת העבודה תהיה זהה בין יוצרי הפרויקט למען איזו המימוןיות של שני הצדדים. הפיתוח מוערך ב-300 שעות.

12.2. ציוד נדרש :

מחשב נייד/נייד אישי עם חיבור לאינטרנט.

.12.3 תוכנות נדרשות:

- IntelliJ IDEA
- JavaFx
- SceneBuilder
- OpenCSV
- Hibernate
- JasperReports
- JavaMail
- MySQL

.12.4 ידע חדש שנדרש ללמידה לצורך ביצוע הפרויקט

JavaFX : נדרש ללמידה כיצד להשתמש ב JavaFx- כדי ליצור את ממשק המשתמש עבור אפליקצייה.

SceneBuilder : יהיה علينا ללמידה כיצד להשתמש ב SceneBuilder - כדי ליצור את ממשק המשתמש הגרפי עבור האפליקציה.

OpenCSV : יהיה علينا ללמידה כיצד להשתמש ב CSV- כדי ליבא ולי יצא מידע אל/ם קבצי CSV ל/ב אפליקציה.

Hibernate : נדרש ללמידה כיצד להשתמש ב Hibernate- כדי לנצל את שכבת הגישה לנוטונים(MySQL) של האפליקציה.

JasperReports : נדרש ללמידה כיצד להשתמש ב JasperReports- כדי להפיק דוחות על עובדים, מנהלים ומידע הקשור אחר באפליקציה.

JavaMail : נדרש ללמידה כיצד להשתמש ב JavaMail- כדי לישם את תקשורת הדוא"ל באפליקציה.

ארQUITטורת שרת-לקוח: יהיה לנו לצלול לעומק ההבנה כיצד ישומי שרת-לקוח פועלם וכיצד לפתח אותם. צריך לדעת לכתוב את הקוד שיתקשר עם השרת דרך הרשות וכו'.

12.5. ספרות ומקורות מידע

מדריכים באינטרנט גוגל ופורומים בנושא:

Q&A, guides from:
stackoverflow.com/
w3schools.com/

Guides & Courses:
youtube.com/

E-Books for additional and deep information(example):
<https://freecomputerbooks.com/The-Free-Java-Book.html>

AI generated tools:
ChatGPT.com

13. תכנית עבודה ושלבים לימוש הפרויקט:

תכנון וayette דרישות/מידע:
צலנו למקומות של דרישות אפליקציה מסווג HRMIS(כגון- שלד, ארכיטקטורה, לוגיקה וכו').
זהינו סיכונים/באים – וטיפלנו בהם.

עיצוב:
יצירת עיצוב למשק המשתמש באמצעות JavaFx&SceneBuilder
הוספנו CSS למחלקות השונות כחלק מהעיצוב.

פיתוח:
הגדרת סביבת פיתוח באמצעות IntelliJ IDEA
פיתוח של צד לקוח באמצעות SceneBuilder ו JavaFx
פיתוח של צד השרת באמצעות ספריות מבוססות של ג'אווה(כגון io .Hibernate (java util, java io) ו .OpenCSV CSV באמצעות ייבוא/ייצוא של נתונים מקובץ CSV באמצעות JasperReports
הטמעת תכונת הדיווח באמצעות JavaMail ע"י .JavaMail

תחזוקה ובדיקה סופית:
בדיקות התוכנה מהאפקטים שונים ידנית כגון- אימות משתמשים, אבטחה ובדיקות שהמערכת פועלת כיחידה אחת ומבצעת הכל כמצופה منها(אינטרציה). הבדיקות כללו בין היתר איתור באגים, בעיות וסידורם.
לבסוף גימור ועטיפה.

14. תכנון הבדיקות שיבוצעו:

14.1. נא פרט בטבלה, בדיקות תהליכיות ברמת משתמש בהן נדרש המערכת לעמוד

תיאור הבדיקה	תיאור הביצוע	תוצאה צפיה	עבר/נכשלה
1. אימות משתמש	המשתמש התחבר בהצלחה.	1. ניסיון כניסה עם משתמשים קיימים. משתמשים לא קיימים.	
	המשתמש לא הצליח להיכנס למערכת וקיבל הודעה שגיאה.	2. ניסיון כניסה עם משתמשים לא קיימים.	
2. יבוא נתונים עובך CSV	האפליקציה צריכה תציג למשתמש חלון בו הוא צריך לבחור את קובץ ה-CSV ליבוא.	1. בחירת אפשרות ליבוא נתונים מקובץ CSV.	
	נתוני עובך מקובץ CSV יבואו בהצלחה לאפליקציה.	2. בחירת קובץ CSV תקין.	
	המשתמש מקבל הודעה שגיאה רלוונטית.	3. בחירת קובץ CSV לא תקין.	
3. הוספה עובך	האפליקציה צריכהבקשת מהמשתמש להזין מידע על עובך ולבחור מחלקה.	1. בחירת אפשרות להוסיף עובך למחלקה.	
	עובד נוסף בהצלחה למחלקה שנבחרה.	2. הצעת פרטי עובך ובחירה מחלקה.	
4. הסרת עובך	האפליקציה צריכה תבקש מהמשתמש לבחור עובך ולאשר הסרה.	1. בחירה אפשרות להסיר עובך.	
	העובד הוסר בהצלחה מהמחלקה.	2. בחר עובך להסרה. ואשר את ההסרה.	
5. הוספה מחלקה	האפליקציה צריכה תבקש להזין פרטי מחלקה.	1. בחירת האפשרות להוסיף מחלקה.	
	מחלקה נוספת בהצלחה.	2. בחירת שמירה של המחלקה החדשה.	
5. הסרת/עדכון מחלקה	האפליקציה צריכה תבקש מהמשתמש לבחור מחלקה ולאשר הסרה/עדכון.	1. בחירה אפשרות להסיר/לעדכן מחלקה.	

	האפליקציה تعدכן את המחלקה בהצלחה.	2. לאחר הצענת נתוני העדכן.	
	האפליקציה מבקשת מהמשתמש לבחור קרייטריונים לדו"ח.	1. בחר באפשרות להפקת דו"ח עבור.	6. הפקת דו"ח עבור
	דו"ח עבור נוצר ומוצג למשתמש.	2. בחר קרייטריונים לדו"ח.	
	האפליקציה מבקשת מהמשתמש לבחור בעובד ולכתוב את המשוב.	1. בחר באפשרות לשЛОוח משוב לעובד.	7. שליחת משוב לעובד באמצעות אימיל
	המשוב נשלח בהצלחה לעובד שנבחר.	2. בחירת העובד ושליחת המשוב.	
	האפליקציה מבקשת מהמשתמש לבחור עובד ולעדכן את המידע שלו.	1. בחר באפשרות לשנות/לעדכן את פרטי העובד הקיימים.	8. שינוי פרטיים לעובד הכוללים שכר, מידע אישי וכו'
	מידע העובד עדכן בהצלחה.	2. בחר עובד ועדכן את המידע שלו.	
	האפליקציה מבקשת מהמשתמש לבחור קובץ CSV או להעלות נתונים ישר מסך הנתונים.	1. בחר באפשרות עדכן משמרות לעובד.	9. הוספה משמרות של עובד עם פרטיים כגון שהתחלת/סיום, תאריך וכו'
	הנתונים ישתקפו בהצלחה בטבלה.	2. בחירה בין הعلاאת הקובץ מהמחשב או טעינת הנתונים ממסך הנתונים.	
	התוכנה שאל אם למחוק את השורה הנוכחית או תעשה ולידציה לעדכן.	1. בחר למחוק/לעדכן שורה של נתונים מטבלת הנתונים.	10. הסרת/עדכן משמרות לעובד
	העדכן/הסרה הושלמו בהצלחה ונשמרו הנקודות CSV והן בסיס נתונים.	2. התוכנה תבדוק שאכן הנתונים תואמים לביקשות.	

14.2. נא פרט בטבלה, מס מיצג של בדיקות ייחודית למודולים המרכזיים בהן נדרשת המערכת לעמוד. (unit test):

מודול	תיאור	ייצוג ייחידת בדיקה
1. אימות	ויריאציה יכולים להיכנס ולגשstag לאפליקציה בדוק התחברות מוצלחת, בדוק התחברות כושלת עם נתונים שגויים.	ויריאציה יכולים להיכנס ולגשstag בדוק יבוא קובץ CSV עם נתונים חוקיים, בדוק יבוא קובץ CSV עם נתונים לא חוקיים.
2. יבוא/ייצוא נתונים	ויריאציה יכולים ליבא ולייצא נתונים אל/מקבצי CSV בדקה לציררת מחלקה, בדקה למחיקת מחלקה, בדקה לשינוי נתונים מחלקה.	ויריאציה יכולים לנהל את נתונים המחלקה שליהם
3. ניהול מחלקה	ויריאציה יכולים לנהל נתונים/עובד/ מחלקות בדקה יצירת עובד, בדקה מהיקת עובד, בדקה לשינוי נתונים עובד.	ויריאציה יכולים לנהל נתונים/עובד/ ראשי מחלקות
4. ניהול עובדים	ויריאציה יכולה להפיק דוחות בהתאם לצרכי המשמש בדקה אם נתונים תקפים, בדקה אם נתונים לא חוקיים.	ויריאציה יכולה להפקיד בדקה אם נתונים תקפים, בדקה אם נתונים לא חוקיים.
5. הפיקט דוחות	ויריאציה יכולים לשלוח חשוב על עובדים דרך האימייל. בדקה תקשורת אימייל.	ויריאציה יכולים לשלוח חשוב על עובדים דרך האימייל.
6. תקשורת		

ן ווּהֶסְבָּרִים כָּלְליִים:

דף כניסה ראשי:

ת.ז. העובד לפיו יכנס למערכת והוא תצהה
איזה תפקיד המשתמש הנ"ל יציג דף כניסה
מתאים

כינוי משתמש :
ת.ז. :
סיסמה :

כניסה **יציאה**

סיסמת המשתמש לכיסיה למערכת
כפתור כניסה למערכת
כפתור יציאה מהאפליקציה

דף כניסה ראשי – שגיאה בהכנסת פרטיים:

כינוי משתמש :
ת.ז. :
סיסמה :

שם משתמש או סיסמה שגויה -
ייתנו 3 ניסיונות סה"כ ואם לא
צלח, המשתמש צריך להמתין 5
דקה עד לניסיון הבא.

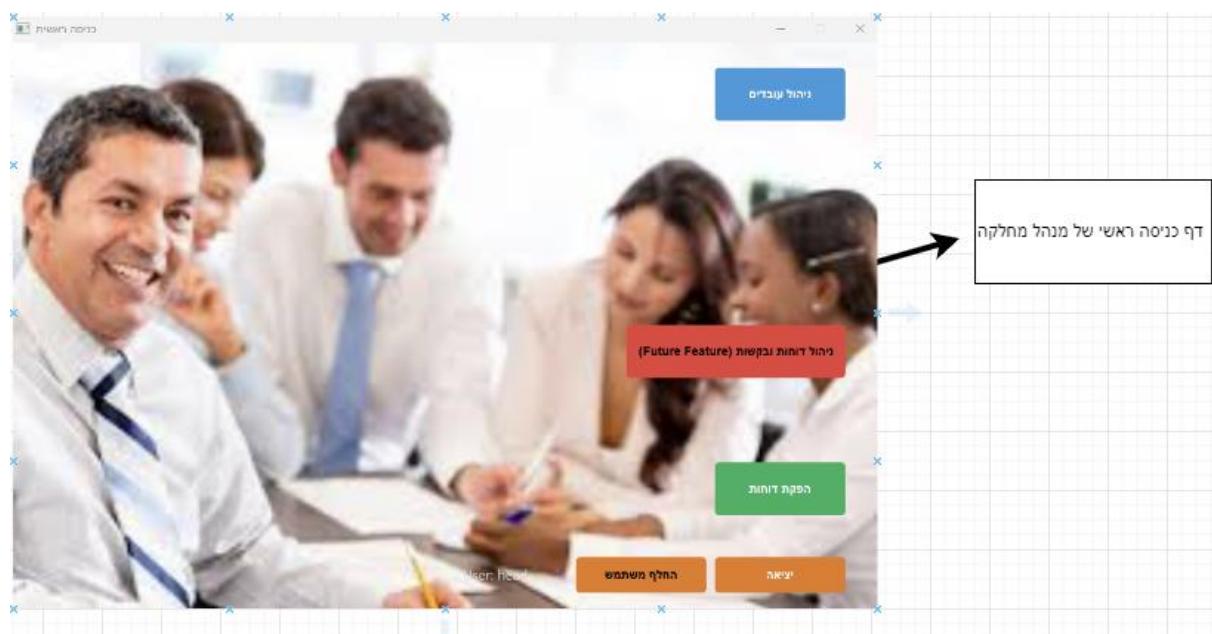
אם משתמש או סיסמה שגויה, ניסיונות שנותה: 2

כניסה **יציאה**

דף כניסה ראשי-מנהל או מנהל בכיר:



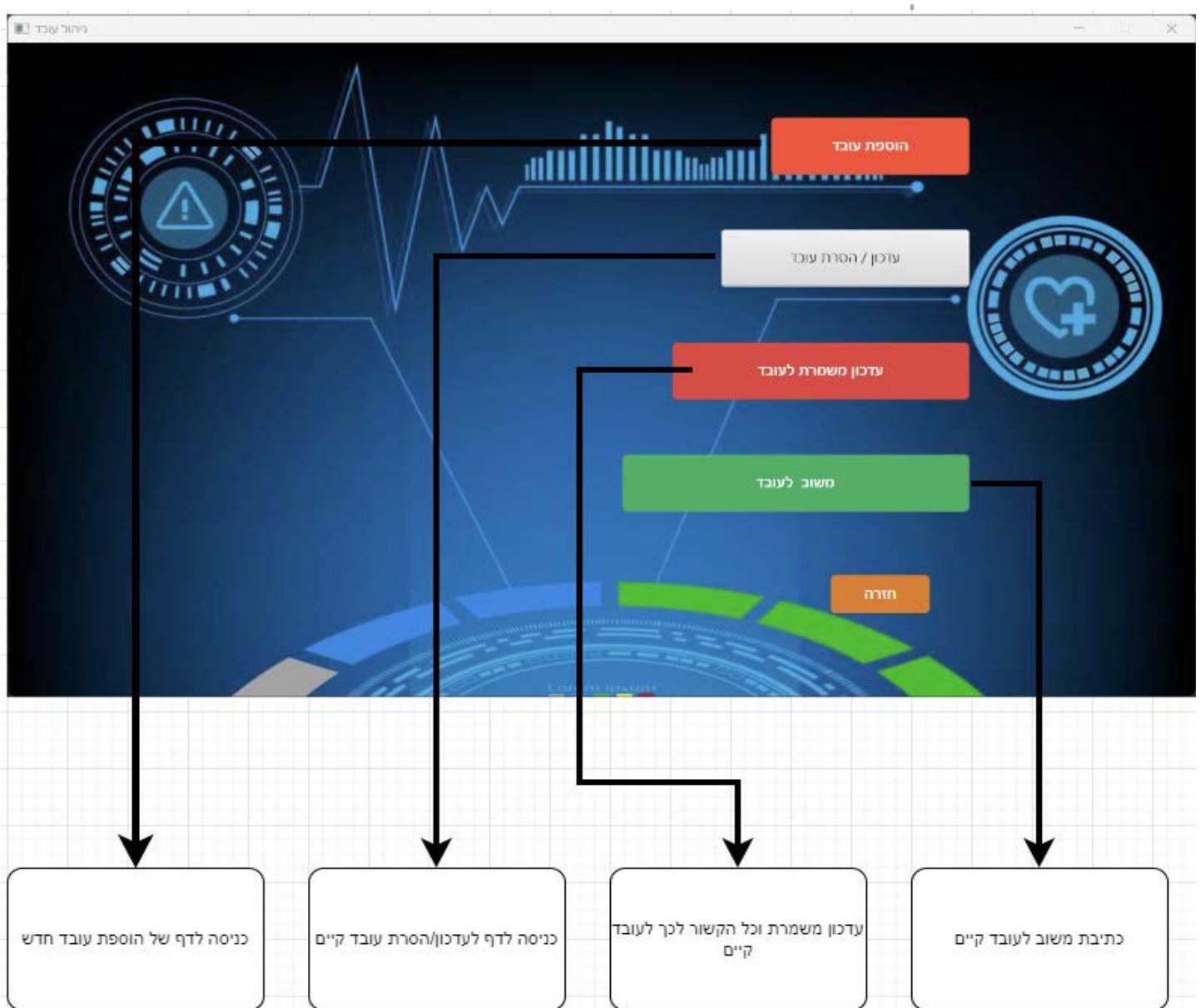
דף כניסה ראשי-מנהל מחלקה:



דף כניסה ראשי-מצירה:



דף ניהול עובדים:



דף הוסף עובד חדש:

כל המוסף בכוכבית הינט פרטן חובת האזנה ע"מ שייה ניתן לשמר בהצלחה.

צורת עובד חדש

הכנסת עובד חדש

* <input type="text"/>	שם פרטי :
* <input type="text"/>	שם משפחה :
* <input type="text"/>	ת.ז. :
* <input type="text"/>	דוא"ל :
<input type="button" value="▼"/>	מגדר :
* <input type="text"/>	מספר טלפון :
<input type="button" value="א.בחר סיסמה"/> <input type="button" value="▼"/>	תפקיד :
<input type="button" value="▼"/>	מחלקה :
<input type="text"/> <input type="button" value="▼"/>	תאריך לידה :
<input type="text"/> <input type="button" value="▼"/>	תאריך תחילת עבודה :
<input type="text"/> <input type="checkbox"/>	שעות :
<input type="text"/> <input type="checkbox"/>	גלובלי :
<input type="text"/>	נסיעות חודשי :
<input type="text"/>	מספר במק' :
<input type="text"/>	מספר סניף :
<input type="text"/>	מספר חשבון :

**ניתן לראות שללא ת.ז(אחד מפרטי החובה) לא יהיה ניתן לשמר
ותופיע שגיאה עם הכוונה של מה לעשות. כמו גם בשאר השדות
אשר ישנה כוכبية בצדם:**

צירט עובד חדש

הכנסתעובד חדש

שם פרטי: עובד
שם משפחה: חדש
ת.ז.:
דוא"ל: sesfa@gmail.comm
מגדר:
מספר טלפון: 0546783728
תפקיד:
מחלקה:
תאריך לידה:
תאריך תחילת עבודה:
שעת:
אלגמי:
נסיעות חודשי:
מספר בנק:
מספר סניף:
מספר חשבון:

נא הכנס ת.ז.

OK

שמירה חזרה

Error

**ת.ז. כבר קיימת במערכת וכאן לא ניתן להוסיףעובד חדש עם אותו
ת.ז – זה Unique לכל עובד:**

יצירת עובד חדש

הכנסת עובד חדש

שם פרטי :	עובד	*
שם משפחה :	חדש	*
ת.ז :	456456456	*
דוא"ל :	sesfa@gmail.commm	*
מגדר :		
מספר טלפון :	0546783728	*
תפקיד :	א. בחר סיסמה	
מחלקה :		
תאריך לידיה :		
תאריך תחילת עבודה :		
שעות :		
גלובלי :		
נסיעות חודשי :		
מספר בנק :		
מספר סניף :		
מספר חשבון :		

OK

Error

ת.ז כבר נמצא במערכת.

שמירה חזרה

לאחר שכל הולידציות אקן נכונות – ניתן לשמור את העובד בהצלחה:

הכנסת עובד חדש

* שם פרטי :	עובד
שם משפחה :	חדר
ת.ז. :	876876876
דוא"ל :	sesfa@gmail.comm
מגדר :	▼
מספר טלפון :	0546783728
תפקיד :	empl...▼
מחלקה :	יעצמים▼
תאריך לידיה :	▼
תאריך תחילת עבודה :	▼
שעת :	▼
גלובלי :	▼
נסיעות חודשי :	▼
מספר בנק :	▼
מספר סניף :	▼
מספר חשבון :	▼

Confirm

נתוני העובד התווספו בהצלחה.

OK

שמירה חזרה

דף עדכון או הסרת עובד:
כאן אנחנו נבחר עובד מהרשימה ולאחר מכן נבחר האם לעדכן את
השדות הקיימות ולוחץ שמירה לעדכונם, או שנבחר בכפטור הרשימה
כדי להסיר את העובד הנבחר.

תחיקת עובד

עדכון/הסרת עובד

בחירה עובד : james bond

*	שם פרטי : james
*	שם משפחה : bond
*	ת.ז. : 456456456
	דוא"ל : dfsdf@fasf.com
	כתובת : <input type="button" value="▼"/>
*	מספר טלפון : 056789654
	תפקיד : <input type="button" value="1234"/> <input type="button" value="secre...▼"/>
88.0	שיעור לשעה : <input checked="" type="checkbox"/>
8000.0	שכר חודשי : <input type="checkbox"/>
0.0	הוצאות חודשי :
0	מספר בנק :
0	מספר סניף :
	מספר חשבון :

הסרת שמירה חזרה

דף עדכון שעות/משמרת עובד:

הזנת ת.ז של עובד ולאחר לחיצה על כפתור חיפוש - יופיע בטבלה הרפיטים הקשורים לת.ז הרשומה אם היא קיימת במערכת

כאן ייצאו כפתורי מחיקה של שורות בעת תוללת נתונים לטבלה

עדכון שעות עובד

הצגת טבלה

ת.ז	תאריך	שם	Delete
			X

הוסף שורה חדשה

הס夙וית שומרה אחרונה

העלאת קובץ שעות

חזר

שמירה

הוסף שורה חדשה לעדכון/הוספה - אם נגמרה שגיאות הנוסה דבויות או מערכתי או הפעול לא החתמים כרטיסי יהיה ניתן להוסף את בשורה החדש וע"י "מחיקת השורה הלא נסנה".

היסטוריית שומרה אחרונה הלקוכה מסך הנתונים כגבוי למקרה של המוקודות קובץ סי-א-ו-ו או שגיאיה הקשורה לקובץ

העלאת קובץ שעות נכון CSV

בעת לחיצה על כפתור "היסטוריית שמירה אחורונה" תציג התוכנה בטבלה

נתונים ממסד הנתונים:

עדכון שעות עבודה

ת.ז.	הוֹן תְּזַעֵּבָד	חפש
678678678	היסטוריית שמירה אחורונה	העלאת קובץ שעות
567567567		
678678678		מחיקה
	הוספה שורה חדשה	

שמירה חזרה

הודעת שגיאת רלוונטיות במקרה והפרטים לא הוזנו יהיה ניסיון שמירה :

עדכון שעות עבודה

ת.ז.	הוֹן תְּזַעֵּבָד	חפש
678678678	היסטוריית שמירה אחורונה	העלאת קובץ שעות
567567567		
678678678		מחיקה
678678678		מחיקה
678678678		מחיקה
234234234		מחיקה
	הוספה שורה חדשה	

שמירה חזרה

Information

נא למלא את העמודות הבאות:

- סדר ساعات
- שעת יציאה
- שעת כניסה
- תאריך

OK

הודעת שגיאת בונגו ניסיון של שמירה של אותו התאריך של אותו בן אדם (ת.ז) שכבר קיימת המערכת, המשמש יctrkr למחוק את השורה השגיאה ולהכנס שורה חדשה עם הפרטים הנכונים ולאחר מכן לשמר בהצלחה:

עדכון שעות עובד

ת. שעות	שם השעות	זמן הספקה	שעת יציאה	שעת כניסה	תאריך	ת.	Delete
9:00		17:00	8:00	19/03/2022	678678678	<input type="button" value="מחיקה"/>	
8:00		8:00	6:00	18/03/2022	678678678	<input type="button" value="מחיקה"/>	
8:00		8:00	6:00	18/03/2022	567567567	<input type="button" value="מחיקה"/>	
10:00		19:00	9:00	18/05/2022	678678678	<input type="button" value="מחיקה"/>	
10:00		19:00	9:00	18/05/2022	234234234	<input type="button" value="מחיקה"/>	
10:00		19:00	9:00	18/05/2022	234234234	<input type="button" value="מחיקה"/>	

ניסיון הכנסת פורמט תאריך שגוי – התוכנה תוציא שגיאה איך למלא נכון את

התאריך:

עדכון שעות עובד

עדכון שעות עובד

תג עובד: חפש חzn tag עובד

סהכ שעות	זמן הפסיקה	שעת יציאה	שעת כניסה	תאריך	תג	Delete
9:00		17:00	8:00	19/03/2022	678678678	<button>מחיקה</button>
8:00		8:00	6:00	18/03/2222	678678678	<button>מחיקה</button>
8:00		8:00	6:00	18/03/2222	567567567	<button>מחיקה</button>
10:00		19:00	9:00	18/05/2222	678678678	<button>מחיקה</button>
10:00		19:00	9:00	18/05/2222	234234234	<button>מחיקה</button>
10:00		19:00	9:00	15	234234234	<button>מחיקה</button>

העלאת קובץ שעות | היסטוריית שמירה אחרונה | חזרה | OK | הוספה שורה חדשה | שמירה | חזרה

Information

פורמט תאריך שגוי, אנא הכנס בפורמט הבא: dd/mm/yyyy

OK

פורמט שעות שגוי – התוכנה תראה שגיאה איך למלא את השעות נכון:

עדכון שעות עובד

עדכון שעות עובד

תג עובד: חפש חzn tag עובד

סהכ שעות	זמן הפסיקה	שעת יציאה	שעת כניסה	תאריך	תג	Delete
9:00		17:00	8:00	19/03/2022	678678678	<button>מחיקה</button>
8:00		8:00	6:00	18/03/2222	678678678	<button>מחיקה</button>
8:00		8:00	6:00	18/03/2222	567567567	<button>מחיקה</button>
10:00		19:00	9:00	18/05/2222	678678678	<button>מחיקה</button>
10:00		19:00	9:00	18/05/2222	234234234	<button>מחיקה</button>
10:00		19:00	5	23/05/2222	234234234	<button>מחיקה</button>

העלאת קובץ שעות | היסטוריית שמירה אחרונה | חזרה | OK | הוספה שורה חדשה | שמירה | חזרה

Information

פורמט שעות שגוי, אנא הכנס בפורמט הבא: hh:mm

OK

חיפוש משמרת/שעות לפי ת.ז ספציפי אשר יוכנו כדלהן:

הכנסת ת.ז של משתמש ואם הוא קיים במערכת אד' ציג את כל הפרטים הרלוונטיים שלו בטבלה

עדכן שעות עבודה

#	ת.ז	Delete
1	678678678	מחיקה
2	678678678	מחיקה
3	678678678	מחיקה

הנילאת נכון שעותם הוסף שורה חדשה הוסף שורה נוספת חזרה שפירה חזרה

עדכן שעות עבודה

אם העלנו קובץ CSV – לא יוכל להשתמש ב"ההיסטוריה שמירה" מכיוון
שהטבלה כבר מלאה ותוצג שגיאה רלוונטית:

מחיקת שורה – וידוא שהמשתמש באמת מעוניין:

מחיקת השורה צלה:

עדכן שעות עובד

עדכון שעות עובד

מ.א.ע.ת. 678678678

[הוספה שורה חדשה](#) [ההיסטוריה שמירה אחרונה](#) [העלאת קובץ שיעות](#)

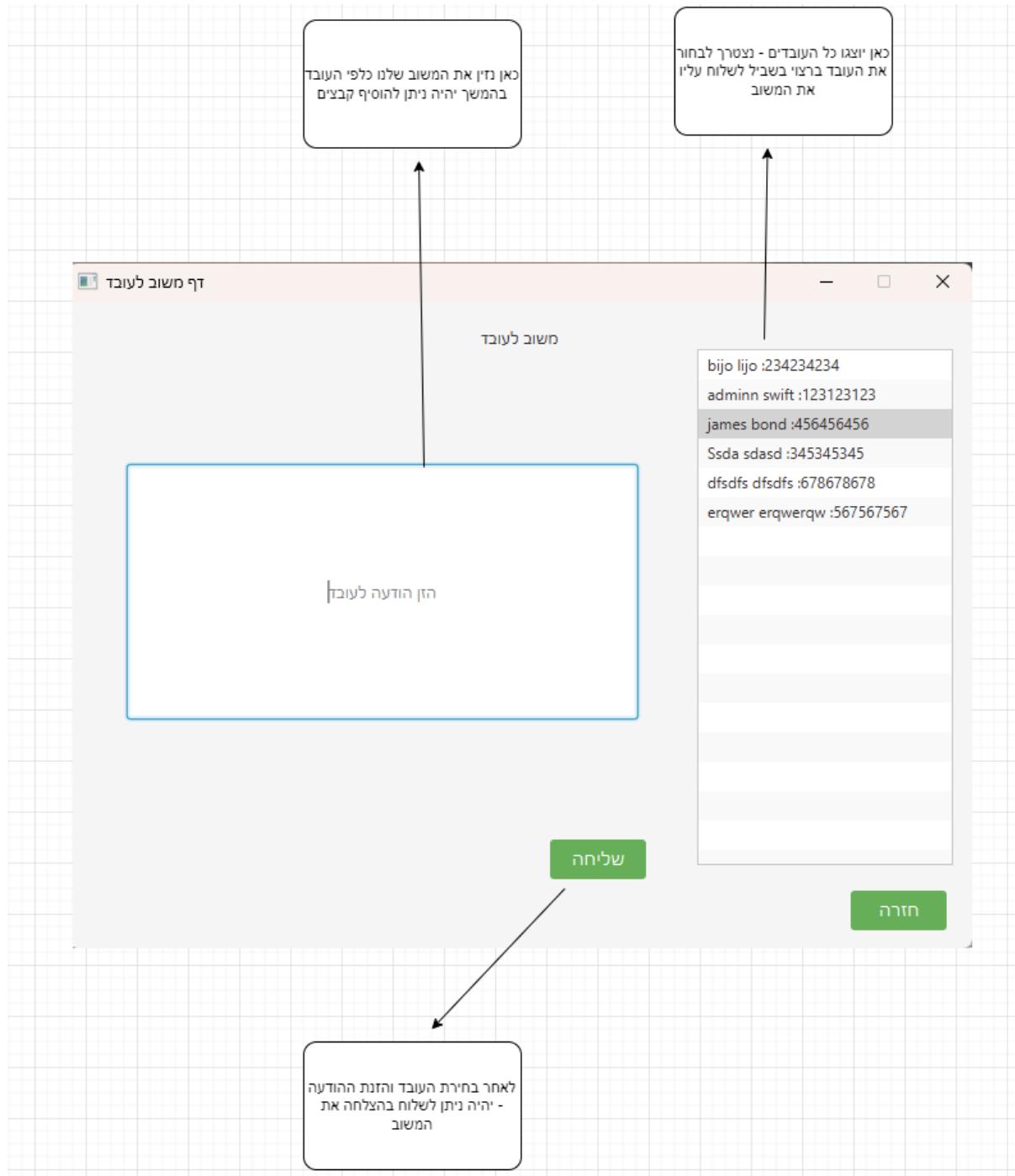
העלאת קובץ שיעות

שמירה

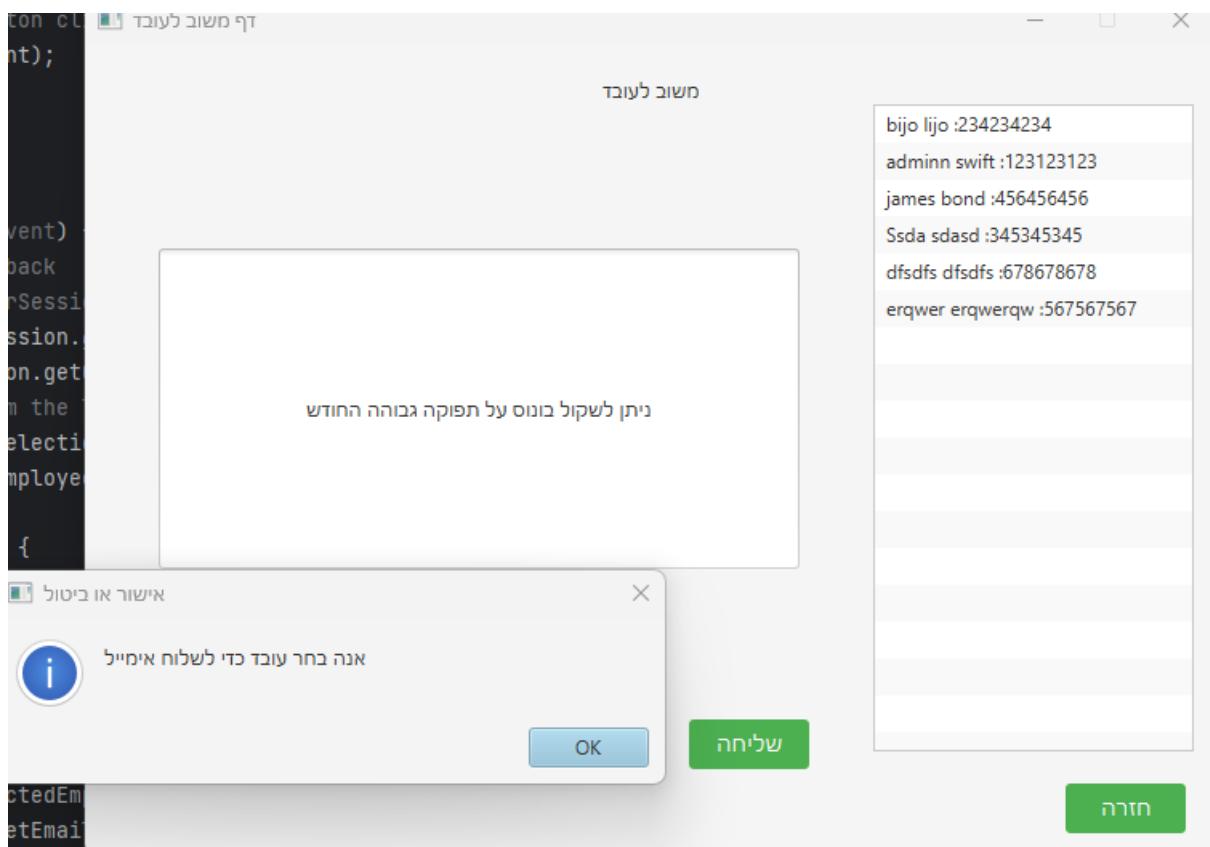
ח'ז

לאחר שכל הולידציות נכונות – יהיה ניתן לשמר את הנתונים לקובץ CSV ולמסד נתונים בהצלחה:

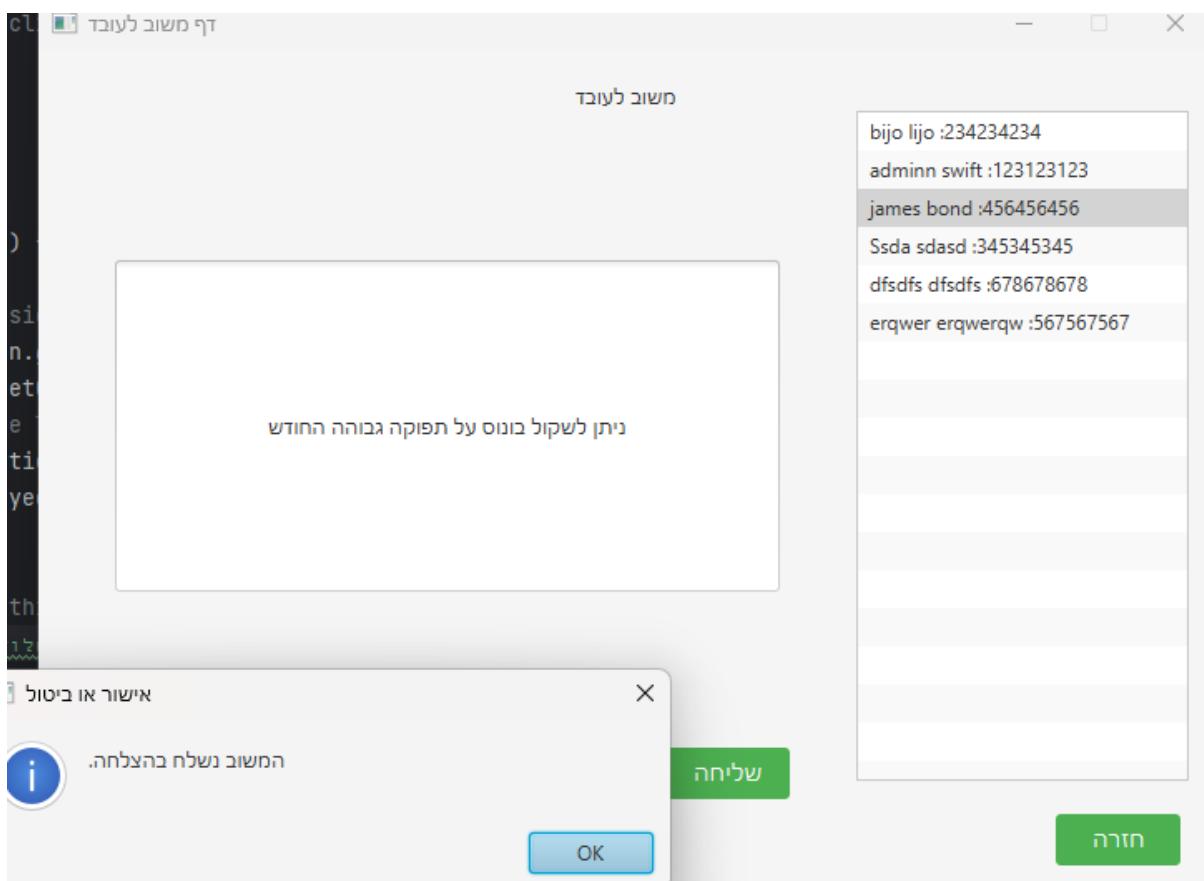
דף מושב לעובד:



הודעה שצרי לבחר עובד כדי לשЛОח מושב:



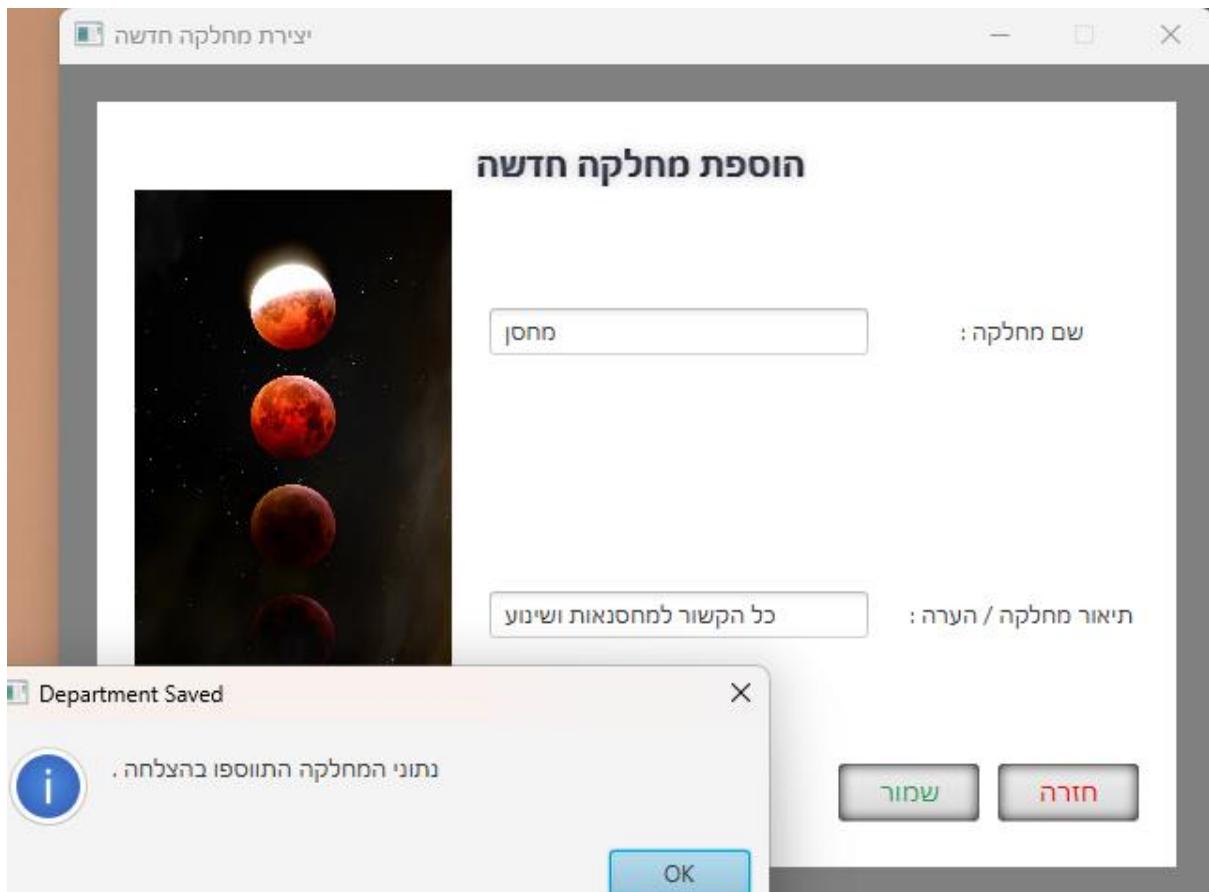
המשוב נשלח בהצלחה לאחר בחירת עובד:



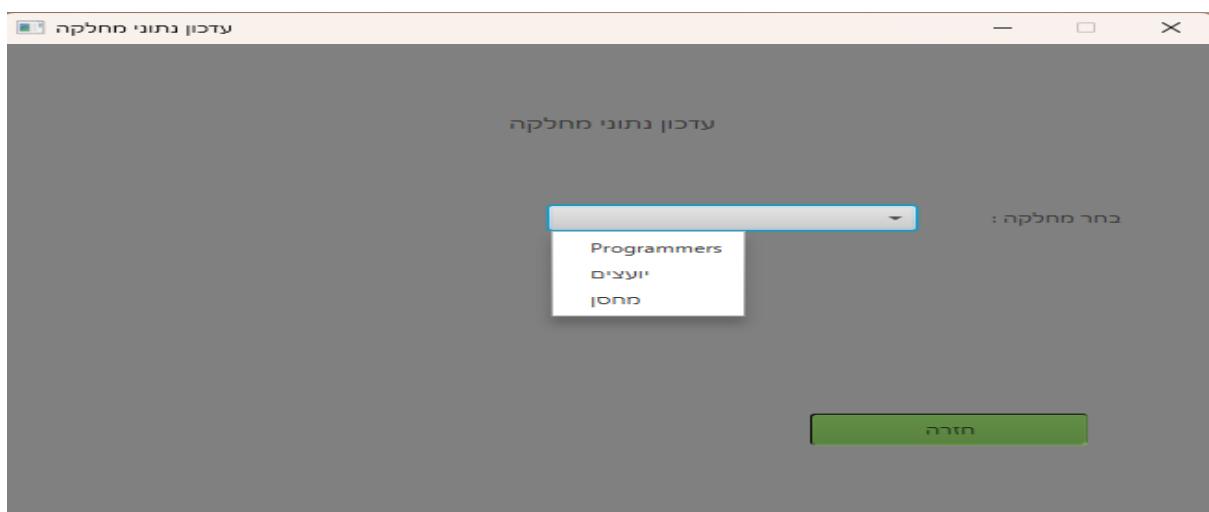
דף ניהול מחלקות:



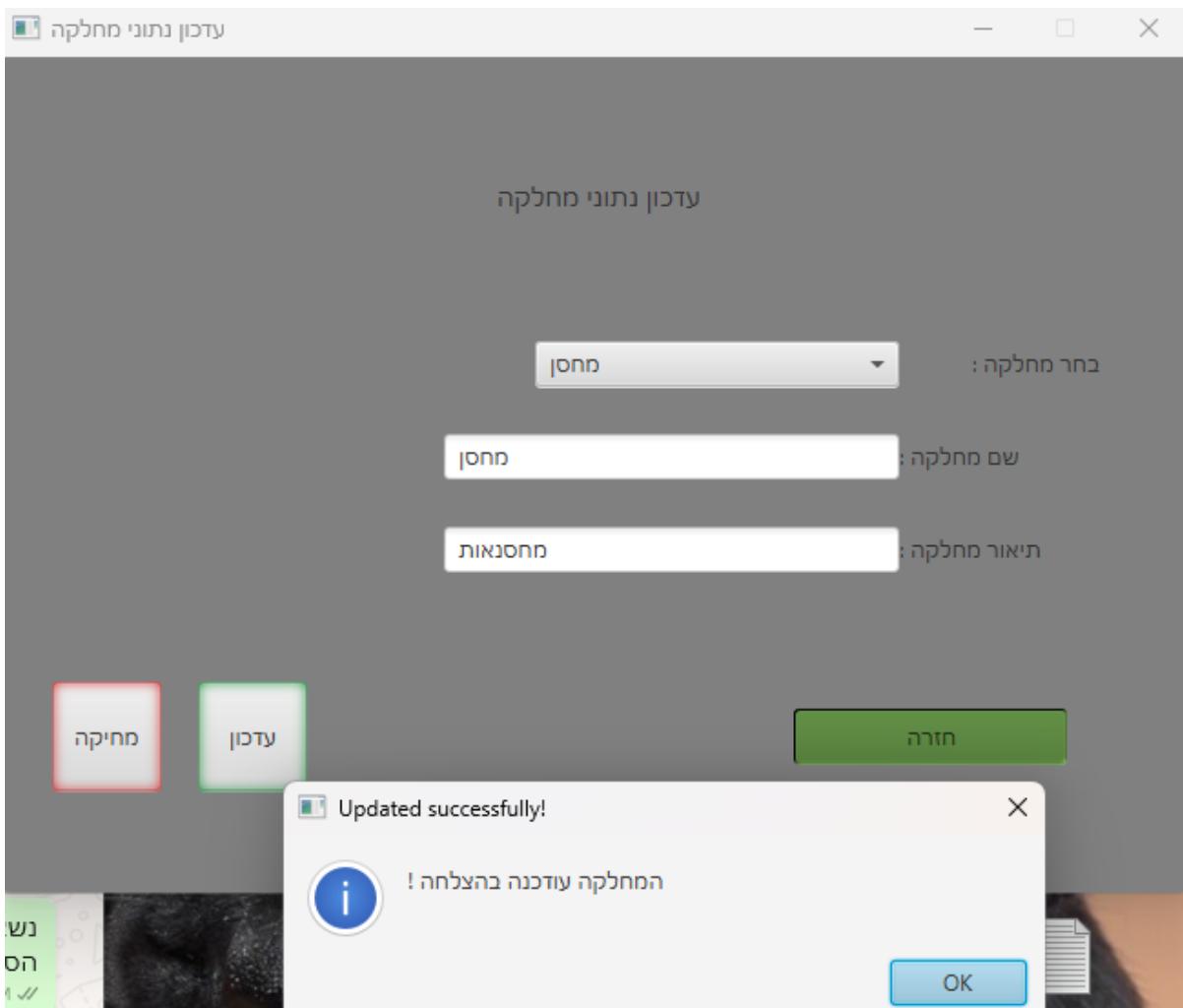
דף הוסף מחלוקת - ניתן שם לחלוקת ותייר (אופציונאל) ונוסיף אותה:



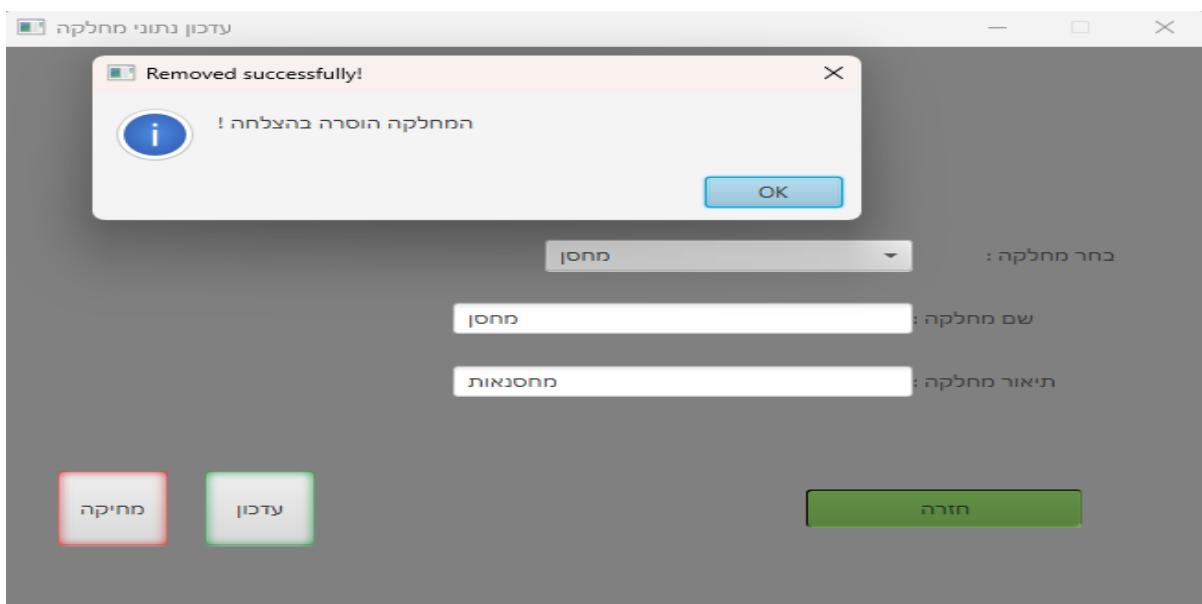
דף עדכון/מחיקת מחלוקת –-can ספציפית נבחר את המחלוקת:



לאחר מכן ניתן את הפרטים לעדכון של המחלקה:

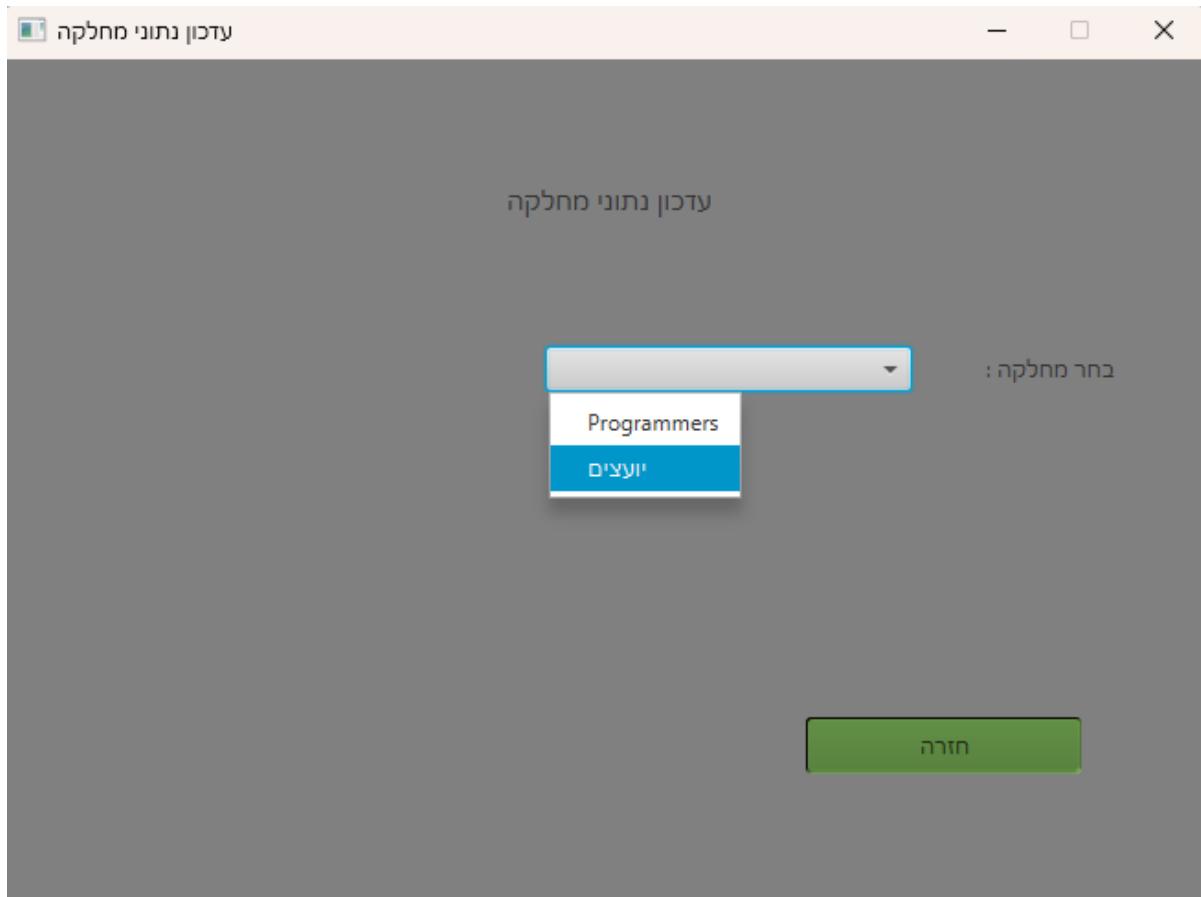


ואם נרצה למחוק את המחלקה אז פשוט נשתמש בכפתור המحיקה:



תוצאה לאחר מחיקת המחלקה – אין אותה כבר לא ברשימה ולא בסוד

הנתונים:



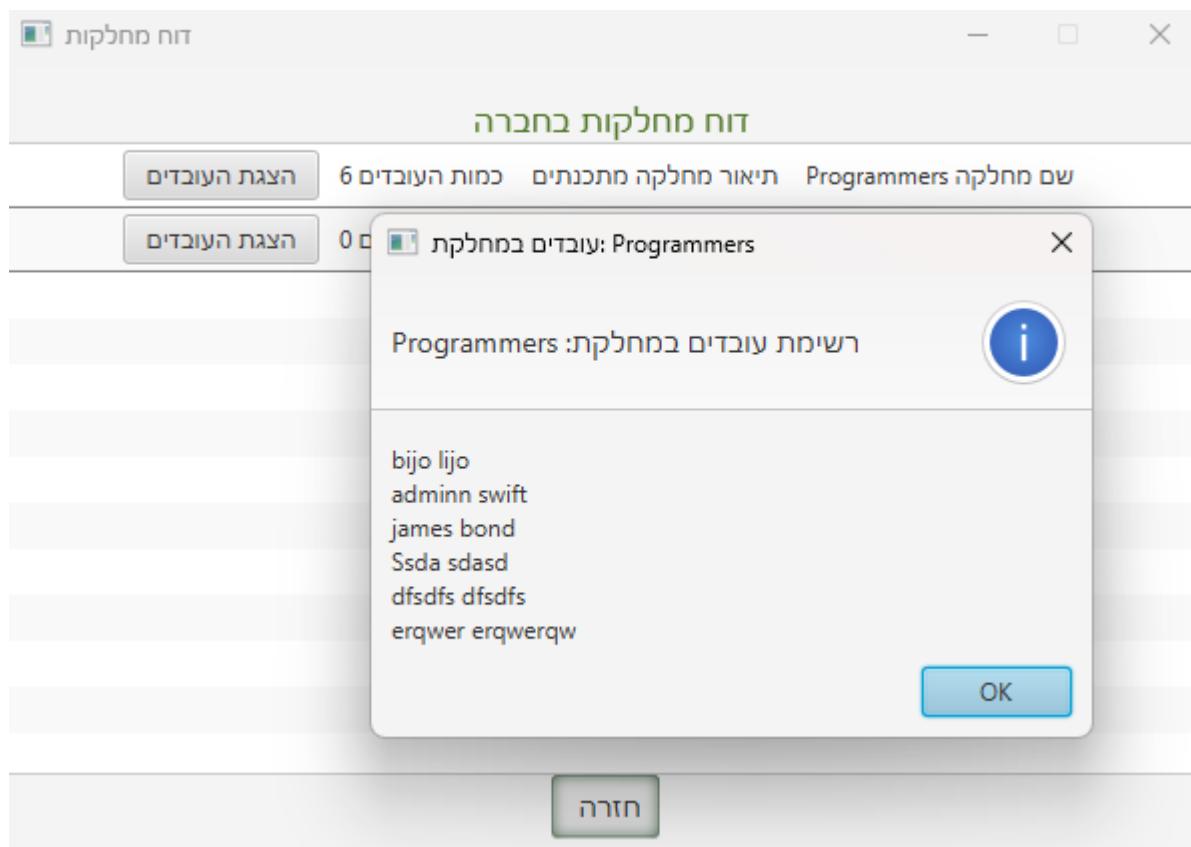
דף דוח מחלקות בחברה – שיציג אילו עובדים שייכים אליה:

דוח מחלקות בחברה

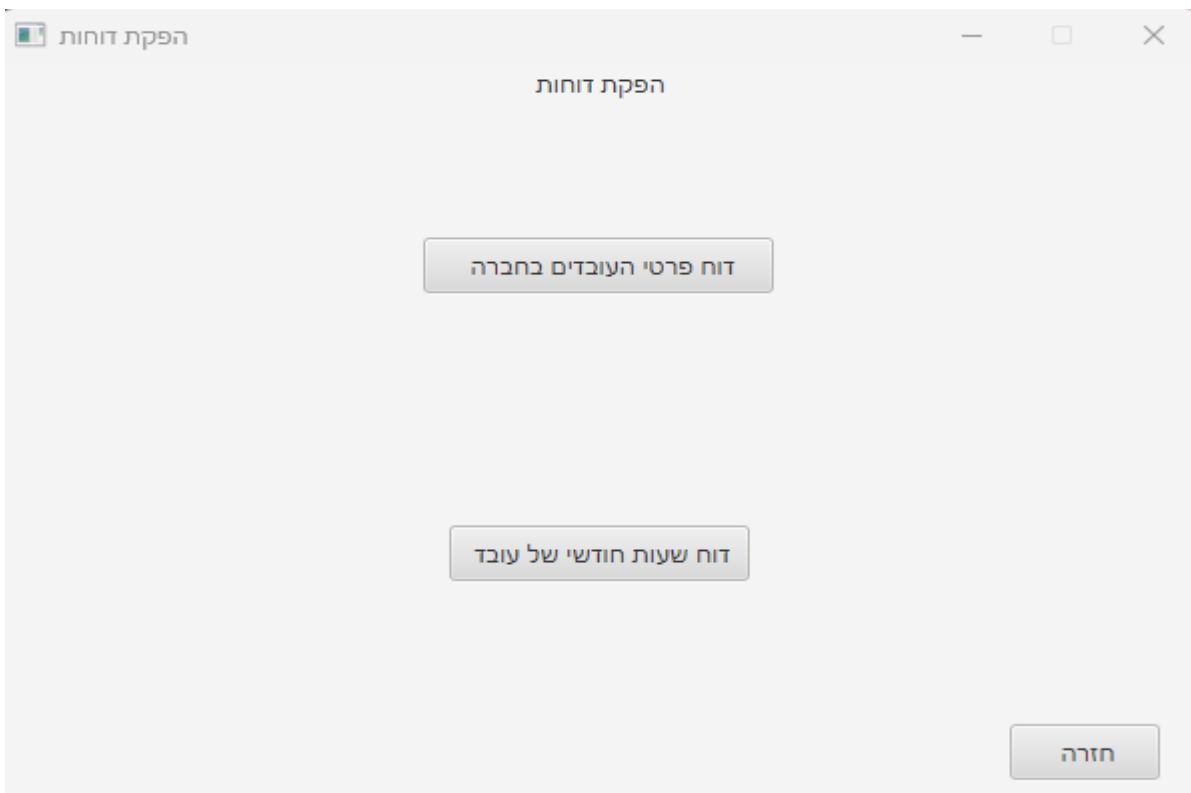
הציגת העובדים	שם מחלקה	תיאור מחלקה	כמות העובדים
יעזים	Programmers	מתכנתים	6
הציגת העובדים	יעזים	يُوصى بـ	0

חזור

לאחר לחיצה על כפתור "הציג עובדים" – כל העובדים במחלקה הנבחרת יוצגו:



דף הפקת דוחות:



דף דוח פרטי העובדים בחברה – אשר יציג פרטים כלליים של כל העובדים בחברה:

asperViewer

100%

HR Pulse

עובד בחברה

מחלקה	מספר טלפון	דוא"ל	מספר נייד	שם משפחה	שם פרטי
Programmers	0.0		0557280193	lijo	bijo
Programmers	0.0	fgfgfg@fasfkh.com	05678929	swift	adminn
Programmers	8000.0	dfsdf@fasf.com	056789654	bond	james
Programmers	0.0	wrtwertwe	45245234	sadasd	Ssda
Programmers	0.0	dflgsdfg	453245	dfsdfs	dfsdfs
Programmers	0.0	dfasdfas	43523452	erqwerqw	erqwer

HR PULSE Report Sunday 04 February

1

Page 1 of 1

דף דוח שעות חודשי של עובד:

הפקת דוחות

בחר חודש :

שם פרטי:	שם משפחה:	טלפון:
biyo	lijo	234234234
adminn	swift	123123123
james	bond	456456456
Ssda	sdasd	345345345
dfsdfs	dfsdfs	678678678
erqwer	erqwerqw	567567567

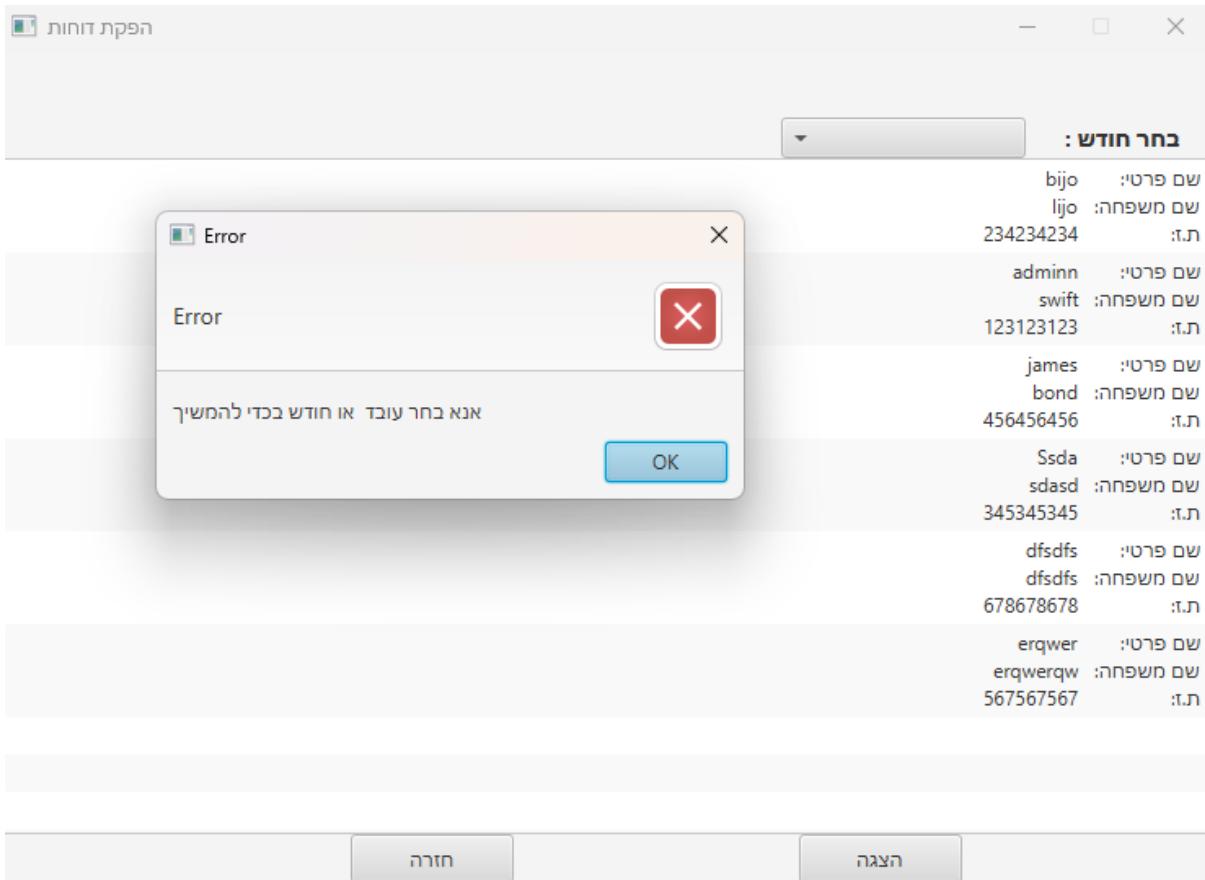
חזרה הצגה

כאן נבחר את החודש בו נרצה להציג את שעות המוכחת שלו

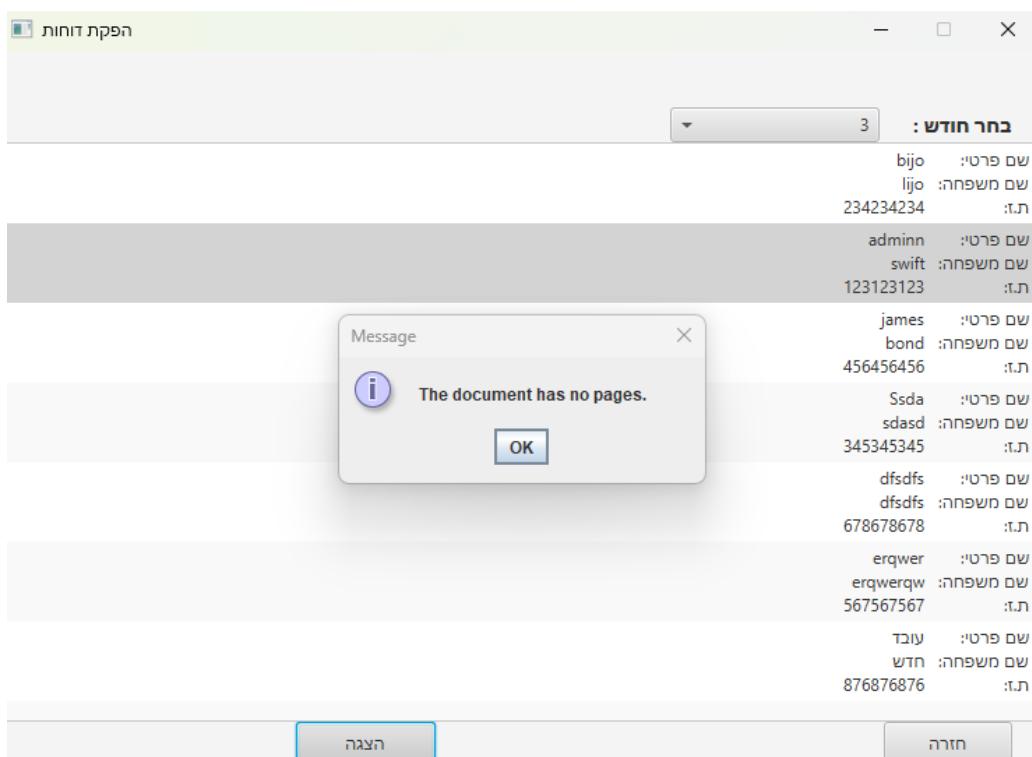
כאן תהיה רשימת העובדים המלאה ממנה נרצה לבחור איזה עובד נרצה להציג את הדוח חודשי שלו

לאחר בחירה של חודש ועובד - לחיצה על כפתור הצג ייתן לנו דוח שעות ותאריכים של אותו החודש של העובד הנבחר

הודעת שגיאה – כי צריך לבחור עובד לפני הפקת הדוח:



הצגת הודעת – הדף ריק מתוכן בעט בחירת חדש אשר העובד לא היה בו ואין נתוניים:



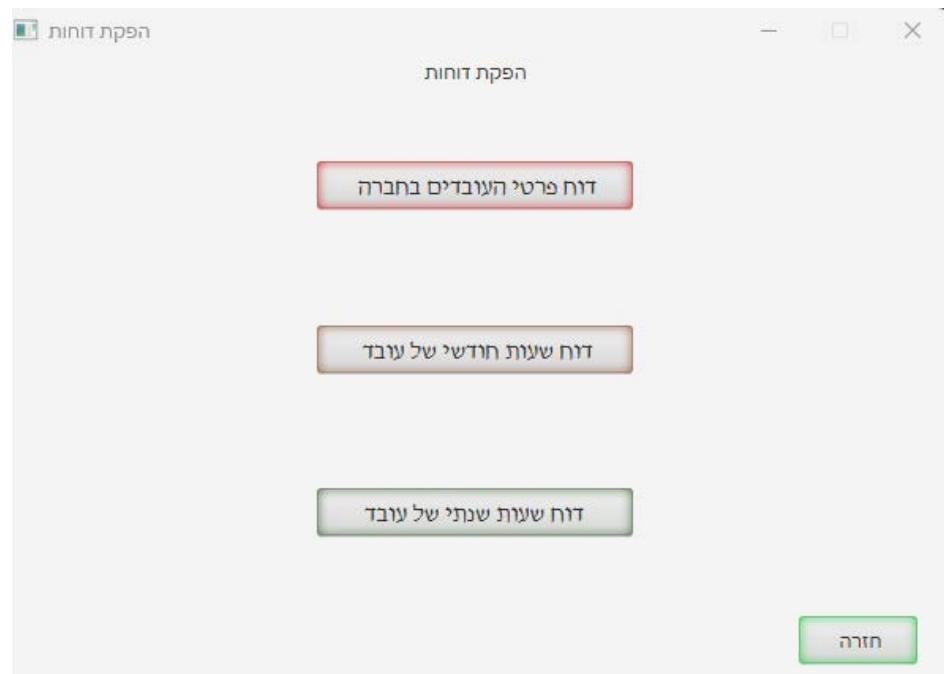
לאחר בחירת עובד עם חדש שנכח בו – יופק הדוח כדלהלן:

The screenshot shows a JasperViewer application window. At the top, there is a small modal dialog titled "הפקת דוחות" (Report Generation) with a dropdown menu set to "5". Below it, a sub-menu titled "בחר חדש" (Select New) is open, showing fields for "שם פרטי" (First Name) with "bijo" and "שם משפחה" (Last Name) with "lijo". The main report area has a title "דוח שעות עובד" (Employee Work Log Report). Below the title is a table with two rows of data. The table has columns: ת.ז. (ID), שם פרטי (First Name), שם משפחה (Last Name), תאריך (Date), תחילת (Start Time), סיום משמרות (End Time), ורכ' שעות (Work Log Type). The data is as follows:

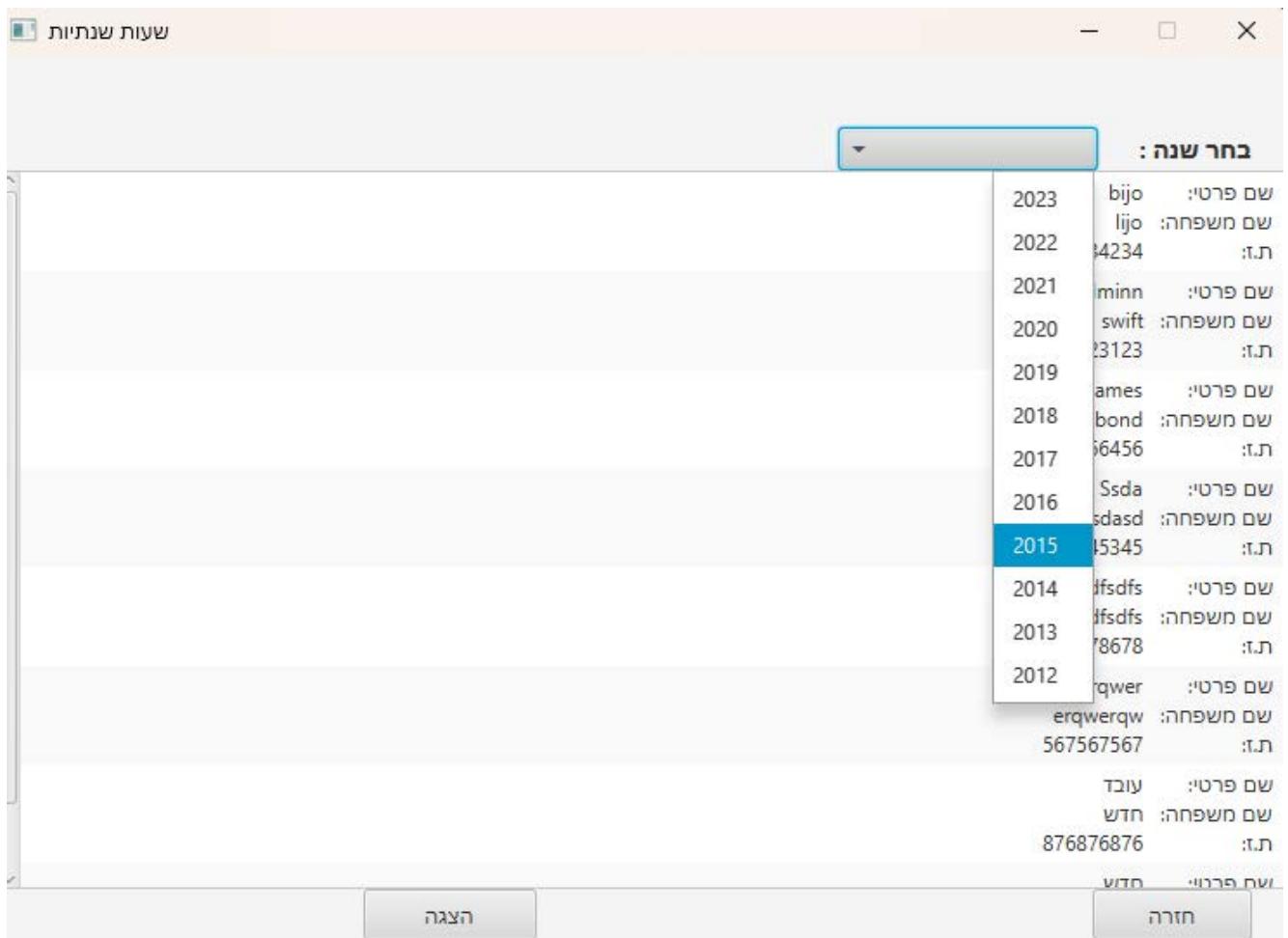
ת.ז.	שם פרטי	שם משפחה	תאריך	תחילת	סיום משמרות	ורכ' שעות
234234234	bijo	lijo	18/05/2222	9:00	19:00	10:00
234234234	bijo	lijo	23/05/2222	9:00	19:00	10:00

At the bottom of the report area, it says "Page 1 of 1".

דף דוח שעות שנתי של עובד



בחירה שנה בה רוצים לראות את נוכחות העובד



הדווח עצמו:

HR PULSE

דוח שעות שנתי לעובד

ת.ז	שם פרטי	שם משפחה	תאריך	התחלה	סיום משמרת	סה'כ שעות
	חדש	בדיקות	23/05/2022	9:00	9:00	9:00



Sunday 25 February

פרטי העובדים בסיס נתונים – ככה ישמרו:

The screenshot shows the MySQL Workbench interface. The top menu bar includes File, Edit, View, Query, Database, Server, Tools, Scripting, and Help. The left sidebar has sections for Schemas, Tables (including employeeShiftData), Columns, and various system objects like Views, Stored Procedures, and Functions. The main area features a 'Query 1 x' tab with the following SQL code:

```
1 #DELETE FROM employeeShiftData
2 #SELECT * FROM employeeShiftData
3 • SELECT * FROM employees
4 #SELECT MAX(id) FROM employeeShiftData
5 #SELECT MIN(id) FROM employeeShiftData
6
7 /*ALTER TABLE employeeShiftData
8 DROP COLUMN comments;
9 */
10
11 /*ALTER TABLE employeeShiftData
12 ADD COLUMN compositeKey VARCHAR(255)
13 */
14
15 #SELECT * FROM departments
16 #SELECT department, employee_id FROM employees
17
18 /*ALTER TABLE employeeShiftData
```

The bottom section displays a 'Result Grid' with the following data:

	id	email	first_name	last_name	bank_info_id	date_of_birth	department	employee_id	employee_role	employment_start_date	hourly_rate	hours_worked	pas
▶	19	bijo	ljo	3	2023-12-19	Programmers	234234234	headOfDepartment	2023-12-28	555.00	NULL	123	
▶	20	fgfgf@fasrh.com	adminn	swift	4	NULL	Programmers	123123123	headOfDepartment	2024-01-02	0.00	NULL	123
▶	21	dfsdf@fasf.com	james	bond	5	NULL	Programmers	456456456	secretary	2023-12-30	88.00	NULL	123
▶	22	wrtwrtwte	Ssda	sdasd	6	NULL	Programmers	345345345	headOfDepartment	2024-01-05	0.00	NULL	123
▶	23	dfgsdfg	dfsdfs	dfsdfs	7	NULL	Programmers	678678678	employee	2024-01-05	0.00	NULL	123
▶	24	dfasdasf	erqwer	erqwerqw	8	NULL	Programmers	567567567	employee	2024-01-05	0.00	NULL	123
◀	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

On the right side, there are buttons for SQLAdditions, Jump to, and context help. The status bar at the bottom shows 'employees 3 x' and 'Context Help Snippets'.

פרטי משמרות העובדים בטבלה בסיס הנתונים – ככה יישמרו:

MySQL Workbench

HR-pulse - Warning - not sup... X

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

pulsedb

- Tables
 - bank_info
 - departments
 - employees
 - Columns
 - Indexes
 - Foreign Keys
 - Triggers
 - employeeshiftdata
 - Columns
 - id
 - total_work_hours
 - break_time
 - employee_id
 - end_of_shift
 - start_of_shift
 - date
 - compositeKey
 - Indexes
 - Foreign Keys
 - Triggers
 - Views
 - Stored Procedures
 - Functions

Query 1 x

1 #DELETE FROM employeeshiftdata

2 • SELECT * FROM employeeshiftdata

3 #SELECT * FROM employees

4 #SELECT MAX(id) FROM employeeshiftdata

5 #SELECT MIN(id) FROM employeeshiftdata

6

7 /*ALTER TABLE employeeShiftData

8 DROP COLUMN comments;

9 */

10

11 /*ALTER TABLE employeeshiftdata

12 ADD COLUMN compositeKey VARCHAR(255)

13 */

14

15 #SELECT * FROM departments

16 #SELECT department, employee_id FROM employees

17

18 /*ALTER TABLE employeeshiftdata

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: |

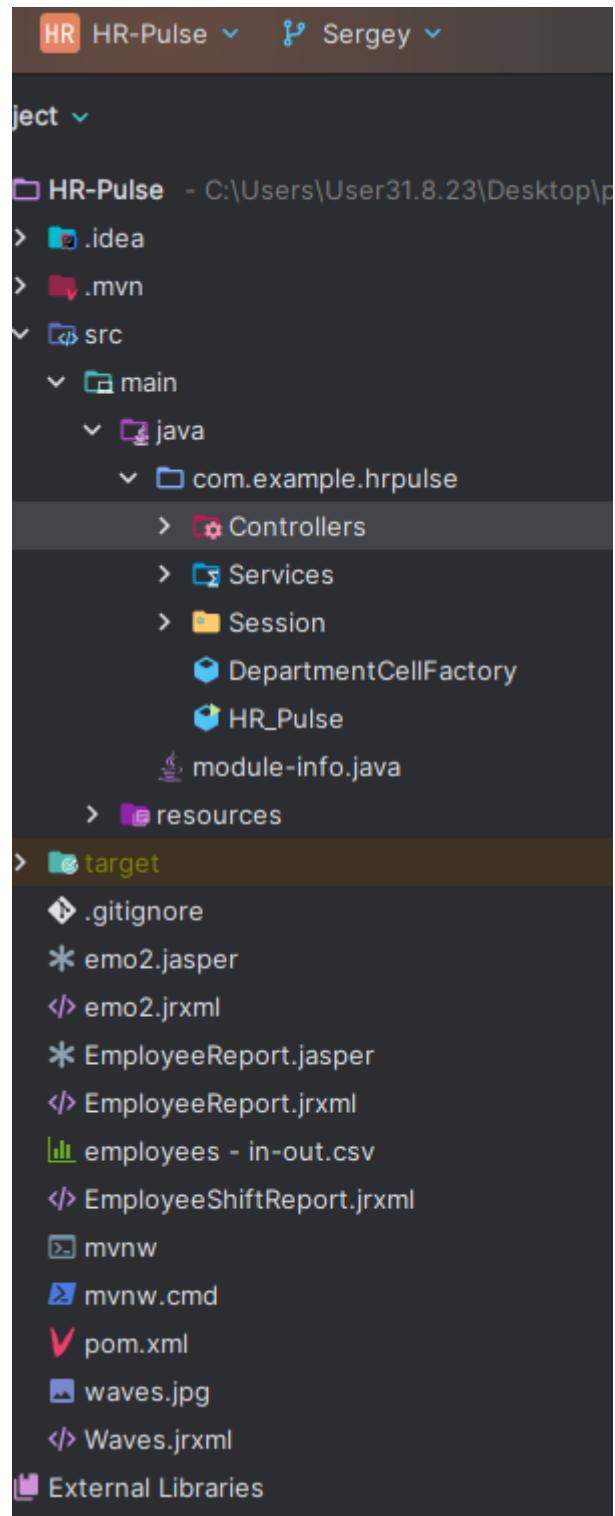
	id	total_work_hours	break_time	employee_id	end_of_shift	start_of_shift	date
166	8:00			678678678	8:00	6:00	18/03/2022
181	8:00			567567567	8:00	6:00	18/03/2022
183	9:00			678678678	17:00	8:00	19/03/2022
185	10:00			234234234	19:00	9:00	18/05/2022
186	10:00			234234234	19:00	9:00	23/05/2022
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Administration Schemas

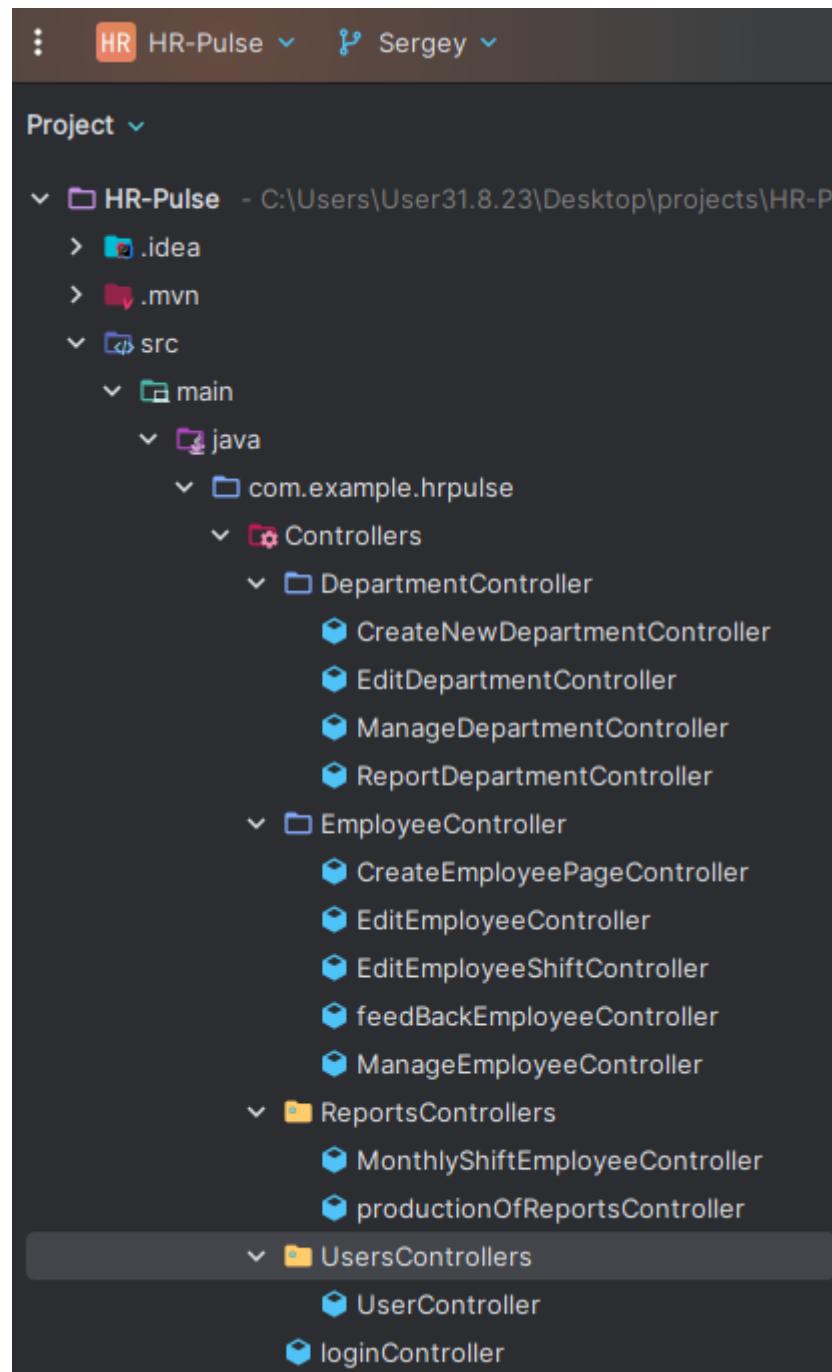
Information

מחלקות, ממשקים וקבצים רלוונטיים נוספים בתוכנה:

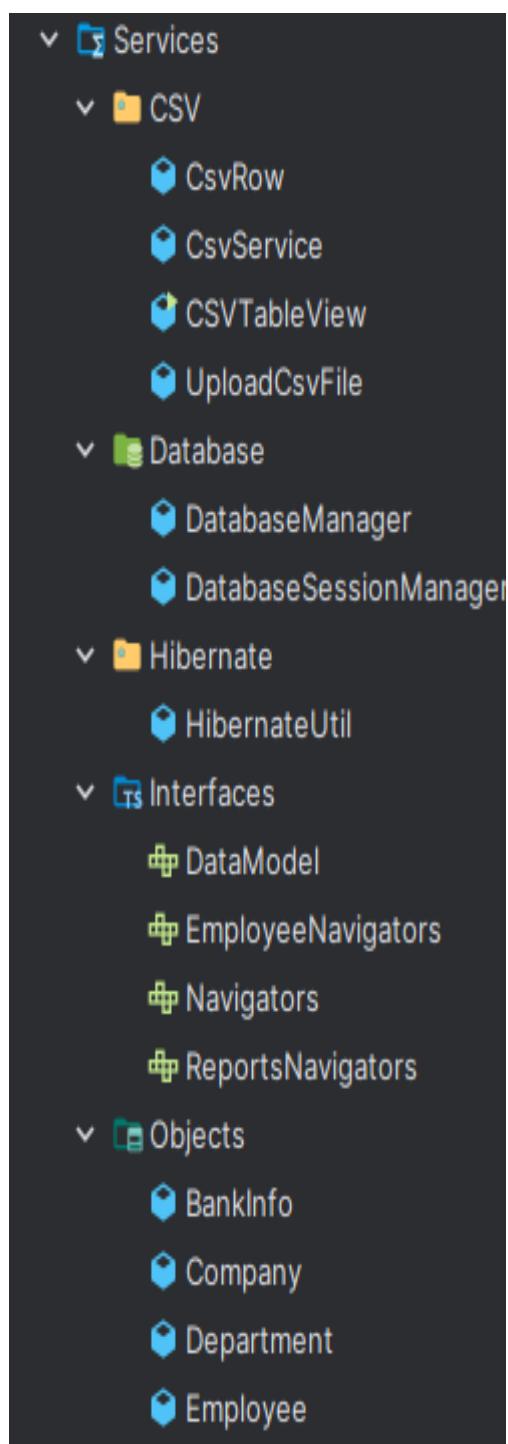
:OVERALL



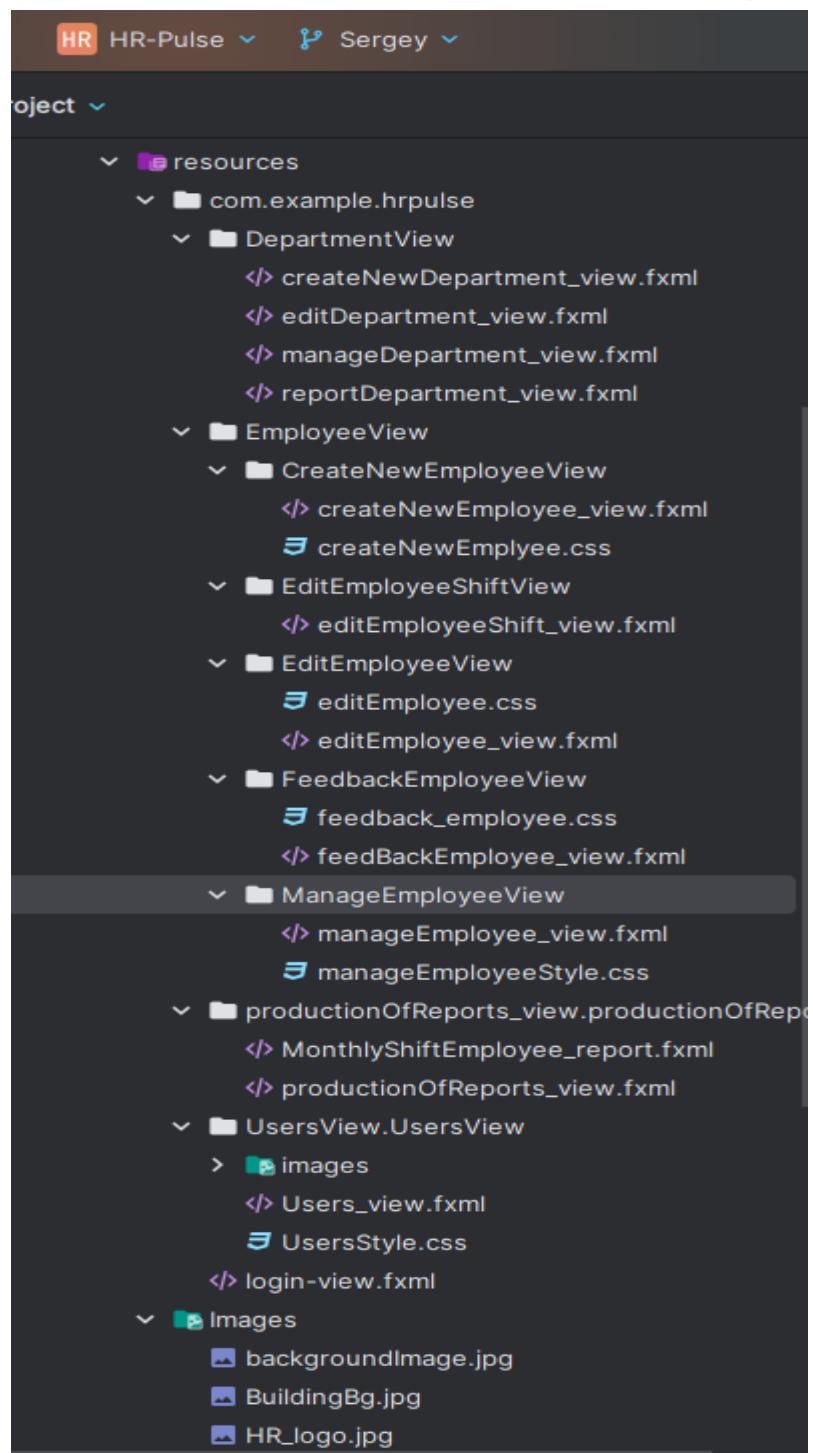
Controllers:



Services:



Recourses and images:



קוד התוכנית:

Main class of the Application HR-Pulse:

```
package com.example.hrpulse;

import javafx.application.Application;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.stage.Stage;
import org.hibernate.SessionFactory;

import java.io.IOException;

/**
 * The `HR_Pulse` class is the main entry point for the HR Pulse
 * application.
 */
public class HR_Pulse extends Application {

    // Hibernate SessionFactory for database operations
    public static SessionFactory sessionFactory;

    // Main method to launch the JavaFX application
    public static void main(String[] args) {
        launch(args);
    }

    // Method called on application start
    @Override
    public void start(Stage stage) throws IOException {
        // Initialize the DatabaseManager when the application starts
        sessionFactory =
            com.example.hrpulse.Services.Database.DatabaseManager.getSessionFactory();

        // Load the login view
        FXMLLoader fxmlLoader = new
        FXMLLoader(HR_Pulse.class.getResource("login-view.fxml"));
        Parent root = fxmlLoader.load();
        Scene scene = new Scene(root);

        // Set up the stage properties
        stage.setResizable(false);
        stage.setTitle("Hr-Pulse");
        stage.setScene(scene);
        stage.centerOnScreen();
        stage.show();

        // Close the SessionFactory when the application exits
        stage.setOnCloseRequest(event -> {

            com.example.hrpulse.Services.Database.DatabaseManager.closeSessionFactory()
            ;
            });
    }
}
```

Hibernate Util:

Hibernate Util:

```
package com.example.hrpulse.Services.Hibernate;

import org.hibernate.SessionFactory;
import org.hibernate.cfg.Configuration;

/**
 * The HibernateUtil class is responsible for managing the Hibernate
SessionFactory.
 */
public class HibernateUtil {

    // The Hibernate SessionFactory instance.
    private static SessionFactory sessionFactory;

    static {
        try {
            // Load configuration from hibernate.cfg.xml
            Configuration configuration = new Configuration().configure();

            // Explicitly add annotated classes to the configuration

            configuration.addAnnotatedClass(com.example.hrpulse.Services.Objects.Employee.class);

            configuration.addAnnotatedClass(com.example.hrpulse.Services.Objects.BankInfo.class);

            configuration.addAnnotatedClass(com.example.hrpulse.Services.Objects.Department.class);

            // Build the SessionFactory
            sessionFactory = configuration.buildSessionFactory();
        } catch (Throwable ex) {
            // Log any initialization error and throw an exception
            System.err.println("Initial SessionFactory creation failed." +
ex);
            throw new ExceptionInInitializerError(ex);
        }
    }

    /**
     * Retrieves the Hibernate SessionFactory.
     *
     * @return The SessionFactory instance.
     */
    public static SessionFactory getSessionFactory() {
        return sessionFactory;
    }

}
```

Session of the user DB interaction:

```
package com.example.hrpulse.Session;

import com.example.hrpulse.Services.Objects.Employee;

/**
 * The `UserSession` class manages the user session and stores information
about the current user.
 */
public class UserSession {

    // Singleton instance
    private static UserSession instance;

    // Current user information
    private Employee currentUser;

    // Private constructor to prevent instantiation
    private UserSession() {
        // Private constructor to prevent instantiation
    }

    /**
     * Gets the singleton instance of UserSession.
     *
     * @return The singleton instance of UserSession.
     */
    public static UserSession getInstance() {
        if (instance == null) {
            instance = new UserSession();
        }
        return instance;
    }

    /**
     * Sets the current user for the session.
     *
     * @param user The Employee representing the current user.
     */
    public void setCurrentUser(Employee user) {
        currentUser = user;
    }

    /**
     * Gets the current user for the session.
     *
     * @return The Employee representing the current user.
     */
    public Employee getCurrentUser() {
        return currentUser;
    }
}
```

DataBase manager:

DataBase manager:

```
package com.example.hrpulse.Services.Database;

import com.example.hrpulse.Services.HibernateUtil;
import com.example.hrpulse.Services.Interfaces.DataModel;
import com.example.hrpulse.Services.Objects.Department;
import com.example.hrpulse.Services.Objects.Employee;
import org.hibernate.HibernateException;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.Transaction;
import org.hibernate.query.Query;

import java.util.Arrays;
import java.util.List;
import java.util.stream.Collectors;

/**
 * The DatabaseManager class handles interactions with the database using
 * Hibernate.
 * It provides methods for saving, deleting, and querying data in the
 * database.
 */
public class DatabaseManager {

    // The Hibernate SessionFactory, responsible for managing database
    sessions.
    private static final SessionFactory sessionFactory =
    HibernateUtil.getSessionFactory();

    /**
     * Gets the Hibernate SessionFactory instance.
     *
     * @return The SessionFactory instance.
     */
    public static SessionFactory getSessionFactory() {
        return sessionFactory;
    }

    /**
     * Closes the Hibernate SessionFactory if it is open.
     */
    public static void closeSessionFactory() {
        if (sessionFactory != null && !sessionFactory.isClosed()) {
            sessionFactory.close();
        }
    }

    // Method to perform database operations after form submission for
    Employee
    public static void performDatabaseOperations(Employee employee) {
        DatabaseSessionManager sessionManager = new
        DatabaseSessionManager(DatabaseManager.getSessionFactory());

        // Save the employee to the database
        boolean saved = sessionManager.saveEmployee(employee);

        if (saved) {
            // Display confirmation
        }
    }
}
```

```

        System.out.println("Employee saved successfully.");
    } else {
        // Display error
        System.out.println("Error saving employee.");
    }
}

// Method to retrieve a list of employees from the database
public static List<Employee> retrieveEmployees() {
    try (Session session = sessionFactory.openSession()) {
        session.beginTransaction();

        // Create an HQL query to select all employees
        Query<Employee> query = session.createQuery("from Employee",
Employee.class);
        List<Employee> employees = query.list();

        // Commit the transaction
        session.getTransaction().commit();
        System.out.println("Retrieved " + employees.size() + " employees.");

        return employees;
    } catch (Exception e) {
        e.printStackTrace();
        System.err.println("Error retrieving employees: " +
e.getMessage());
        return null;
    }
}

// Method to perform database operations after form submission for
Department
public static void performDatabaseOperations(Department department) {
    DatabaseSessionManager sessionManager = new
DatabaseSessionManager(DatabaseManager.getSessionFactory());

    // Save the department to the database
    boolean saved = sessionManager.saveDepartment(department);

    if (saved) {
        // Display confirmation
        System.out.println("Departments saved successfully.");
    } else {
        // Display error
        System.out.println("Error saving Department.");
    }
}

// Method to perform database operations after form submission for
Department (update or remove)
public static void performDatabaseOperations(Department department,
boolean update) {
    DatabaseSessionManager sessionManager = new
DatabaseSessionManager(DatabaseManager.getSessionFactory());

    if (update) {
        // Update the department in the database
        boolean updated = sessionManager.updateDepartment(department);

        if (updated) {

```

```

        // Display confirmation
        System.out.println("Department updated successfully.");
    } else {
        // Display error
        System.out.println("Error updating Department.");
    }
} else {
    // Remove the department from the database
    boolean removed = sessionManager.removeDepartment(department);

    if (removed) {
        // Display confirmation
        System.out.println("Department removed successfully.");
    } else {
        // Display error
        System.out.println("Error removing Department.");
    }
}

// Method to retrieve a list of departments from the database
public static List<Department> retrieveDepartments() {
    try (Session session = sessionFactory.openSession()) {
        // Begin a transaction
        session.beginTransaction();

        // Create an HQL query to select all departments
        Query<Department> query = session.createQuery("from
Department", Department.class);

        // Execute the query and get the list of departments
        List<Department> departments = query.list();

        // Commit the transaction
        session.getTransaction().commit();

        return departments;
    } catch (Exception e) {
        e.printStackTrace();
        return null;
    }
}

// Method to perform database operations after form submission for
Employee (update or remove)
public static void employeePDO(Employee selectedEmployee, boolean
update) {
    DatabaseSessionManager sessionManager = new
DatabaseSessionManager(DatabaseManager.getSessionFactory());

    if (update) {
        // Update the employee in the database
        boolean updated =
sessionManager.updateEmployee(selectedEmployee);

        if (updated) {
            // Display confirmation
            System.out.println("Employee updated successfully.");
        } else {
            // Display error
            System.out.println("Error updating employee.");
        }
    }
}

```

```

        }
    } else {
        // Remove the employee from the database
        boolean removed =
sessionManager.removeEmployee(selectedEmployee);

        if (removed) {
            // Display confirmation
            System.out.println("Employee removed successfully.");
        } else {
            // Display error
            System.out.println("Error removing Employee.");
        }
    }
}

/**
 * Saves CSV data to the specified database table.
 *
 * @param tableName The name of the database table.
 * @param csvData The CSV data to be saved.
 */
public static void saveCSVDataToDatabase(String tableName,
List<String[]> csvData) {
    try (Session session = sessionFactory.openSession()) {
        Transaction transaction = null;
        try {
            transaction = session.beginTransaction();

            // Iterate through CSV data and insert new records if they
don't exist
            for (String[] rowData : csvData) {
                String employeeId = rowData[5];
                String date = rowData[4];

                if (!isDataExists(sessionFactory, tableName,
employeeId, date)) {
                    String insertQuery =
generateInsertQuery(tableName);
                    executeInsertQuery(session, insertQuery, rowData);
                }
            }

            transaction.commit();
        } catch (HibernateException e) {
            handleDatabaseException(transaction, e);
        }
    }
}

/**
 * Executes an insert query with the provided data.
 *
 * @param session The Hibernate session.
 * @param insertQuery The insert query.
 * @param rowData The data to be inserted.
 */
private static void executeInsertQuery(Session session, String
insertQuery, String[] rowData) {
    try {

```

```

        Query query = session.createNativeQuery(insertQuery);

        // Set the parameters for the query
        query.setParameter("totalWorkHours", rowData[0]);
        query.setParameter("breakTime", rowData[1]);
        query.setParameter("exitHour", rowData[2]);
        query.setParameter("startHour", rowData[3]);
        query.setParameter("dateTable", rowData[4]);
        query.setParameter("employeeId", rowData[5]);

        query.executeUpdate();
    } catch (HibernateException e) {
        throw new RuntimeException("Error executing insert query", e);
    }
}

/**
 * Deletes a row from the specified database table.
 *
 * @param tableName The name of the database table.
 * @param employeeId The employee ID of the row to be deleted.
 * @param date The date of the row to be deleted.
 */
public static void deleteRowFromDatabase(String tableName, String
employeeId, String date) {
    try (Session session = sessionFactory.openSession()) {
        Transaction transaction = null;
        try {
            transaction = session.beginTransaction();

            String deleteQuery = "DELETE FROM " + tableName + " WHERE
employee_id = :employee_id AND date = :date";
            Query<?> query = session.createNativeQuery(deleteQuery);
            query.setParameter("employee_id", employeeId);
            query.setParameter("date", date);

            query.executeUpdate();

            transaction.commit();
        } catch (Exception e) {
            handleTransactionException(transaction, e);
        }
    }
}

/**
 * Checks if data with the given employee ID and date exists in the
specified table.
 *
 * @param sessionFactory The Hibernate SessionFactory.
 * @param tableName The name of the database table.
 * @param employeeId The employee ID to check.
 * @param date The date to check.
 * @return True if data exists, false otherwise.
 */
public static boolean isDataExists(SessionFactory sessionFactory,
String tableName, String employeeId, String date) {
    try (Session session = sessionFactory.openSession()) {
        String checkExistenceQuery = "SELECT 1 FROM " + tableName + "
WHERE employee_id = :employeeId AND date = :date";
        Query query = session.createNativeQuery(checkExistenceQuery);

```

```

        query.setParameter("employeeId", employeeId);
        query.setParameter("date", date);
        return query.uniqueResult() != null;
    } catch (HibernateException e) {
        handleHibernateException(e);
        return false;
    }
}

/**
 * Generates an insert query for the specified table.
 *
 * @param tableName The name of the database table.
 * @return The insert query.
 */
private static String generateInsertQuery(String tableName) {
    return "INSERT INTO " + tableName +
           " (total_work_hours, break_time, end_of_shift,
start_of_shift, date, employee_id) " +
           "VALUES (:totalWorkHours, :breakTime, :exitHour,
:startHour, :dateTable, :employeeId)";
}

/**
 * Handles exceptions related to database operations.
 *
 * @param transaction The database transaction.
 * @param e             The exception.
 */
public static void handleDatabaseException(Transaction transaction,
HibernateException e) {
    handleTransactionException(transaction, e);
}

/**
 * Handles exceptions related to transactions.
 *
 * @param transaction The database transaction.
 * @param e             The exception.
 */
private static void handleTransactionException(Transaction transaction,
Exception e) {
    if (transaction != null && transaction.isActive()) {
        transaction.rollback();
    }
    handleHibernateException(e);
}

/**
 * Handles general Hibernate exceptions.
 *
 * @param e The exception.
 */
public static void handleHibernateException(Exception e) {
    e.printStackTrace(); // Log or handle the exception appropriately
}

/**
 * Checks if the specified database table is empty.
 *
 * @param tableName The name of the database table.

```

```

        * @return True if the table is empty, false otherwise.
    */
    public static boolean isEmpty(String tableName) {
        try (Session session = sessionFactory.openSession()) {
            String countQuery = "SELECT COUNT(*) FROM " + tableName;
            Query<Number> query = session.createNativeQuery(countQuery);
            Number count = query.getSingleResult();
            return count.longValue() == 0L;
        }
    }

    /**
     * Loads the last saved data from the specified table.
     *
     * @param tableName The name of the database table.
     * @param clazz      The class of the DataModel.
     * @param <T>         The type of the DataModel.
     * @return A list of DataModel objects representing the last saved
data.
    */
    public static <T extends DataModel> List<T> loadLastSavedData(String
tableName, Class<T> clazz) {
        try (Session session = sessionFactory.openSession()) {
            String selectQuery = "SELECT total_work_hours, break_time,
end_of_shift, start_of_shift, date, employee_id " +
                "FROM " + tableName + " ORDER BY total_work_hours DESC,
break_time DESC, " +
                    "end_of_shift DESC, start_of_shift DESC, date DESC,
employee_id DESC";
            Query<Object[]> query = session.createNativeQuery(selectQuery);
            List<Object[]> resultList = query.getResultList();
            return convertToObjectList(resultList, clazz);
        }
    }

    /**
     * Loads the last saved data for a specific employee from the specified
table.
     *
     * @param tableName  The name of the database table.
     * @param clazz      The class of the DataModel.
     * @param employeeId The employee ID to filter the data.
     * @param <T>         The type of the DataModel.
     * @return A list of DataModel objects representing the last saved data
for the specified employee.
    */
    public static <T extends DataModel> List<T>
loadLastSavedDataByEmployeeId(String tableName, Class<T> clazz, String
employeeId) {
        try (Session session = sessionFactory.openSession()) {
            String selectQuery = "SELECT total_work_hours, break_time,
end_of_shift, start_of_shift, date, employee_id " +
                "FROM " + tableName + " WHERE employee_id =
:employeeId " +
                    "ORDER BY total_work_hours DESC, break_time DESC, " +
                        "end_of_shift DESC, start_of_shift DESC, date DESC,
employee_id DESC";
            Query<Object[]> query = session.createNativeQuery(selectQuery);
            query.setParameter("employeeId", employeeId);
            List<Object[]> resultList = query.getResultList();
            return convertToObjectList(resultList, clazz);
        }
    }
}

```

```

        }

    /**
     * Converts a list of object arrays to a list of DataModel objects.
     *
     * @param resultList The list of object arrays.
     * @param clazz      The class of the DataModel.
     * @param <T>         The type of the DataModel.
     * @return A list of DataModel objects.
     */
    private static <T extends DataModel> List<T>
convertToObjectList(List<Object[]> resultList, Class<T> clazz) {
    return resultList.stream()
        .map(row -> {
            try {
                // Create an instance of the DataModel and
                initialize it with CSV data
                T instance =
                clazz.getDeclaredConstructor().newInstance();
                instance.initializeFromCsvData(Arrays.copyOf(row,
row.length, String[].class));
                return instance;
            } catch (Exception e) {
                throw new RuntimeException("Error creating instance
of " + clazz.getSimpleName(), e);
            }
        })
        .collect(Collectors.toList());
}
}

```

**Provided alongside a DB Session manager so we separate it to
maintain better:**

```

package com.example.hrpulse.Services.Database;

import com.example.hrpulse.Services.Objects.Department;
import com.example.hrpulse.Services.Objects.Employee;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.Transaction;
import org.hibernate.query.Query;

import java.util.List;

/**
 * The DatabaseSessionManager class manages database sessions for saving,
updating, and removing Employee and Department entities.
 */
public class DatabaseSessionManager {

    // The Hibernate SessionFactory, responsible for managing database
sessions.

```

```
private final SessionFactory sessionFactory;

/**
 * Constructs a DatabaseSessionManager with the provided
SessionFactory.
 *
 * @param sessionFactory The Hibernate SessionFactory.
 */
public DatabaseSessionManager(SessionFactory sessionFactory) {
    this.sessionFactory = sessionFactory;
}

/**
 * Saves an Employee entity to the database.
 *
 * @param employee The Employee entity to be saved.
 * @return True if the operation is successful, false otherwise.
 */
public boolean saveEmployee(Employee employee) {
    try (Session session = sessionFactory.openSession()) {
        Transaction transaction = session.beginTransaction();

        try {
            session.save(employee);
            transaction.commit();
            return true;
        } catch (Exception e) {
            transaction.rollback();
            e.printStackTrace();
        }
    }

    return false;
}

/**
 * Saves a Department entity to the database.
 *
 * @param department The Department entity to be saved.
 * @return True if the operation is successful, false otherwise.
 */
public boolean saveDepartment(Department department) {
    try (Session session = sessionFactory.openSession()) {
        Transaction transaction = session.beginTransaction();

        try {
            session.save(department);
            transaction.commit();
            return true;
        } catch (Exception e) {
            transaction.rollback();
            e.printStackTrace();
        }
    }

    return false;
}

/**
 * Updates a Department entity in the database.
 *
```

```

        * @param department The Department entity to be updated.
        * @return True if the operation is successful, false otherwise.
    */
public boolean updateDepartment(Department department) {
    try (Session session = sessionFactory.openSession()) {
        Transaction transaction = session.beginTransaction();

        try {
            session.update(department);
            transaction.commit();
            return true;
        } catch (Exception e) {
            transaction.rollback();
            e.printStackTrace();
        }
    }

    return false;
}

/**
 * Removes a Department entity from the database.
 *
 * @param department The Department entity to be removed.
 * @return True if the operation is successful, false otherwise.
 */
public boolean removeDepartment(Department department) {
    try (Session session = sessionFactory.openSession()) {
        Transaction transaction = session.beginTransaction();

        try {
            session.delete(department);
            transaction.commit();
            return true;
        } catch (Exception e) {
            transaction.rollback();
            e.printStackTrace();
        }
    }

    return false;
}

/**
 * Updates an Employee entity in the database.
 *
 * @param selectedEmployee The Employee entity to be updated.
 * @return True if the operation is successful, false otherwise.
 */
public boolean updateEmployee(Employee selectedEmployee) {
    try (Session session = sessionFactory.openSession()) {
        Transaction transaction = session.beginTransaction();

        try {
            session.update(selectedEmployee);
            transaction.commit();
            return true;
        } catch (Exception e) {
            transaction.rollback();
            e.printStackTrace();
        }
    }
}

```

```

        }

        return false;
    }

    /**
     * Removes an Employee entity from the database.
     *
     * @param employee The Employee entity to be removed.
     * @return True if the operation is successful, false otherwise.
     */
    public boolean removeEmployee(Employee employee) {
        try (Session session = sessionFactory.openSession()) {
            Transaction transaction = session.beginTransaction();

            try {
                session.delete(employee);
                transaction.commit();
                return true;
            } catch (Exception e) {
                transaction.rollback();
                e.printStackTrace();
            }
        }

        return false;
    }
}

```

Our Object classes: bankInfo:

```

package com.example.hrpulse.Services.Objects;

import javax.persistence.*;

@Entity
@Table(name = "bank_info")
public class BankInfo {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(name = "id")
    private int id;

    @Column(name = "bankSneefCode")
    private int bankSneefCode;
    @Column(name = "bankNumber")
    private int bankNumber;
    @Column(name = "account_number")
    private String accountNumber;

    public BankInfo() {
        // Default no-argument constructor required by Hibernate
    }

    public BankInfo(String bankName, int bankSneefCode, int bankNumber,
String accountNumber) {

```

```

        this.bankSneefCode = bankSneefCode;
        this.bankNumber = bankNumber;
        this.accountNumber = accountNumber;
    }

    public int getId() {
        return id;
    }

    public int getBankSneefCode() {
        return bankSneefCode;
    }

    public void setBankSneefCode(int bankSneefCode) {
        this.bankSneefCode = bankSneefCode;
    }

    public int getBankNumber() {
        return bankNumber;
    }

    public void setBankNumber(int bankNumber) {
        this.bankNumber = bankNumber;
    }

    public String getAccountNumber() {
        return accountNumber;
    }

    public void setAccountNumber(String accountNumber) {
        this.accountNumber = accountNumber;
    }
}

```

Company:

```

package com.example.hrpulse.Services.Objects;
import java.util.ArrayList;
import java.util.List;

public class Company {
    private static Company myCompany;
    private List<Department> departments= new ArrayList<>();
    private List<Employee>employees=new ArrayList<>();
    private Company() {
    }
    public static Company getMyCompany(){
        if (myCompany == null) {
            // Initialize myCompany if it's null
            myCompany = new Company();
        }
        return myCompany;
    }

    public void addDepartment(Department department) {
        departments.add(department);
    }
    public void addEmployee(Employee employee){
        employees.add(employee);
    }
}

```

```

    }
    public Department getDepartmentByName(String departmentName) {
        for (Department department : departments) {
            if
(department.getDepartmentName().equalsIgnoreCase(departmentName)) {
                return department;
            }
        }
        return null; // Department not found
    }
}

```

Department:

```

package com.example.hrpulse.Services.Objects;
import javax.persistence.*;
import java.util.List;
@Entity
@Table(name = "departments") // Specify the actual database table name
public class Department {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(name ="department_id")
    private int departmentId;
    @Column(name = "department_name")
    private String departmentName;
    @Column(name = "description" )
    private String description;
    @OneToMany(mappedBy = "department", fetch = FetchType.EAGER)
    private List<Employee> employees;

    public Department() {
    }

    public Department(int departmentId, String departmentName,
List<Employee> employees) {
        this.departmentId = departmentId;
        this.departmentName = departmentName;
        this.employees = employees;
    }

    public int getDepartmentId() {
        return departmentId;
    }

    public void setDepartmentId(int departmentId) {
        this.departmentId = departmentId;
    }

    public String getDepartmentName() {
        return departmentName;
    }

    public void setDepartmentName(String departmentName) {
        this.departmentName = departmentName;
    }

    public void setEmployees(List<Employee> employees) {
        this.employees = employees;
    }
}

```

```

        public String getDescription() {
            return description;
        }

        public void setDescription(String description) {
            this.description = description;
        }

        public void addEmployee(Employee employee) {
            employees.add(employee);
        }

        public List<Employee> getEmployees() {
            // Return the list of employees in this department
            return employees;
        }

        @Override
        public String toString() {
            return "Department{" +
                "departmentId=" + departmentId +
                ", departmentName='" + departmentName + '\'' +
                ", description='" + description + '\'' +
                ", employees=" + employees +
                '}';
        }
    }
}

```

Employee:

```

package com.example.hrpulse.Services.Objects;

import java.time.LocalDate;
import java.time.LocalDateTime;
import java.util.Date;
import javax.persistence.*;

@Entity
@Table(name = "employees")
public class Employee {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(name = "id")
    private Integer id;

    @Column(name = "employee_id")
    private Integer employeeId;

    @Column(name = "first_name")
    private String firstName;

    @Column(name = "last_name")
    private String lastName;

    @Column(name = "password")
    private String password;

    @Column(name = "date_of_birth")
    private LocalDate dateOfBirth;
}

```

```

@Column(name = "email")
private String email;

@Column(name = "phone_number")
private String phoneNumber;

@Column(name = "gender")
private String gender;

@Column(name = "employee_role")
private String employeeRole;

@Column(name = "department")
private String department;

@OneToOne(cascade = CascadeType.ALL)
@JoinColumn(name = "bank_info_id")
private BankInfo bankInfo;

@Column(name = "hourly_rate")
private Double hourlyRate;

@Column(name = "employment_start_date")
private Date employmentStartDate;

@Column(name = "hours_worked")
private Integer hoursWorked;

@Column(name = "isHourly")
private Boolean isHourly;

@Column(name = "salaryToTravel")
private Double salaryToTravel;

@Column(name = "salaryPerMonth")
private Double salaryPerMonth;

@Transient
private String username;

public Employee() {
    this.bankInfo = new BankInfo();
}

public Employee(String firstName, String lastName, String email, String
phoneNumber, String password, String username) {
    this.firstName=firstName;
    this.lastName=lastName;
    this.email=email;
    this.phoneNumber=phoneNumber;
    this.password=password;
    this.username = username;
}

public Employee(String firstName, String email, String phoneNumber,
String password) {

    this.firstName=firstName;
    this.email=email;
}

```

```
        this.phoneNumber=phoneNumber;
        this.password=password;
        this.employeeRole = employeeRole;
    }

    public Employee(String firstName, String email, String phoneNumber, String
password, String employeeRole){
        this.firstName=firstName;
        this.email=email;
        this.phoneNumber=phoneNumber;
        this.password=password;
        this.employeeRole =employeeRole;
    }

    public boolean isHourly() {
        return isHourly;
    }

    public void setHourly(boolean hourly) {
        isHourly = hourly;
    }

    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public int getEmployeeId() {
        return employeeId;
    }

    public void setEmployeeId(int employeeId) {
        this.employeeId = employeeId;
    }

    public String getFirstName() {
        return firstName;
    }

    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }

    public String getLastName() {
        return lastName;
    }

    public void setLastName(String lastName) {
        this.lastName = lastName;
    }

    public String getPassword() {
        return password;
    }

    public void setPassword(String password) {
        this.password = password;
    }
```

```
}

public LocalDate getDateOfBirth() {
    return dateOfBirth;
}
public double getSalaryToTravel() {
    return salaryToTravel;
}

public void setSalaryToTravel(double salaryToTravel) {
    this.salaryToTravel = salaryToTravel;
}
public void setDateOfBirth(LocalDate dateOfBirth) {
    this.dateOfBirth = dateOfBirth;
}

public String getEmail() {
    return email;
}

public void setEmail(String email) {
    this.email = email;
}

public String getPhoneNumber() {
    return phoneNumber;
}

public void setPhoneNumber(String phoneNumber) {
    this.phoneNumber = phoneNumber;
}

public String getEmployeeRole() {
    return employeeRole;
}

public void setEmployeeRole(String employeeRole) {
    this.employeeRole = employeeRole;
}

public String getDepartment() {
    return department;
}

public void setDepartment(String department) {
    this.department = department;
}

public BankInfo getBankInfo() {
    return bankInfo;
}

public String getGender() {
    return gender;
}

public void setGender(String gender) {
    this.gender = gender;
}

public void setBankInfo(BankInfo bankInfo) {
```

```

        this.bankInfo = bankInfo;
    }

    public double getHourlyRate() {
        return hourlyRate;
    }

    public void setHourlyRate(double hourlyRate) {
        this.hourlyRate = hourlyRate;
    }

    public Date getEmploymentStartDate() {
        return employmentStartDate;
    }

    public void setEmploymentStartDate(Date employmentStartDate) {
        this.employmentStartDate = employmentStartDate;
    }

    public int getHoursWorked() {
        return hoursWorked;
    }

    public void setHoursWorked(int hoursWorked) {
        this.hoursWorked = hoursWorked;
    }

    public double getSalaryPerMonth() {
        return salaryPerMonth;
    }

    public void setSalaryPerMonth(double salaryPerMonth) {
        this.salaryPerMonth = salaryPerMonth;
    }

}

```

Main Interfaces:

```

package com.example.hrpulse.Services.Interfaces;

import javafx.event.ActionEvent;
import javafx.fxml.FXMLLoader;
import javafx.scene.Node;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.stage.Stage;

import java.io.IOException;
import java.util.Objects;

public interface EmployeeNavigators {
    //-----This interface is handling the
    navigation between the employee page -----

    // ----- navigate to the main employee view page -----
    default void navigateToManageEmployeePage(ActionEvent event) throws

```

```

IOException {
    Parent ManageEmployeePageViewParent =
FXMLLoader.load(Objects.requireNonNull(getClass().getResource("/com/example
/hrpulse/EmployeeView/ManageEmployeeView/manageEmployee_view.fxml")));
    Scene ManageEmployeePageViewScene = new
Scene(ManageEmployeePageViewParent);
    Stage mainStage = (Stage) ((Node)
event.getSource()).getScene().getWindow();
    mainStage.setScene(ManageEmployeePageViewScene);
    mainStage.setTitle("עורך נתונים");
    mainStage.centerOnScreen();
    mainStage.show();
}

//-----
-----
    default void navigateToCreateEmployeePage(ActionEvent event) throws
IOException {
    Parent createEmployeeViewParent =
FXMLLoader.load(Objects.requireNonNull(getClass().getResource("/com/example
/hrpulse/EmployeeView/CreateNewEmployeeView/createNewEmployee_view.fxml")));
;
    Scene createEmployeeViewScene = new
Scene(createEmployeeViewParent);
    Stage mainStage = (Stage) ((Node)
event.getSource()).getScene().getWindow();
    mainStage.setScene(createEmployeeViewScene);
    mainStage.setTitle("יצירת נתונים חדש");
    mainStage.centerOnScreen();
    mainStage.show();
}

    default void navigateToEditEmployeePage(ActionEvent event) throws
IOException {
    Parent removeEmployeeViewParent =
FXMLLoader.load(Objects.requireNonNull(getClass().getResource("/com/example
/hrpulse/EmployeeView/EditEmployeeView/editEmployee_view.fxml")));
    Scene removeEmployeeViewScene = new
Scene(removeEmployeeViewParent);
    Stage mainStage = (Stage) ((Node)
event.getSource()).getScene().getWindow();
    mainStage.setScene(removeEmployeeViewScene);
    mainStage.setTitle("הסרת נתונים");
    mainStage.centerOnScreen();
    mainStage.show();
}

    default void navigateToEditEmployeeShiftPage(ActionEvent event) throws
IOException {
    Parent editEmployeeShiftViewParent =
FXMLLoader.load(Objects.requireNonNull(getClass().getResource("/com/example
/hrpulse/EmployeeView/EditEmployeeShiftView/editEmployeeShift_view.fxml")));
;
    Scene editEmployeeShiftViewScene = new
Scene(editEmployeeShiftViewParent);
    Stage mainStage = (Stage) ((Node)
event.getSource()).getScene().getWindow();
    mainStage.setScene(editEmployeeShiftViewScene);
    mainStage.setTitle("הסרת שフト");
    mainStage.centerOnScreen();
    mainStage.show();
}

```

```

        }

        default void navigateToFeedbackEmployeePage(ActionEvent event) throws
IOException {
    Parent FeedbackEmployeeViewParent =
FXMLLoader.load(Objects.requireNonNull(getClass().getResource("/com/example
/hrpulse/EmployeeView/FeedbackEmployeeView/feedBackEmployee_view.fxml")));
    Scene FeedbackEmployeeViewScene = new
Scene(FeedbackEmployeeViewParent);
    Stage mainStage = (Stage) ((Node)
event.getSource()).getScene().getWindow();
    mainStage.setScene(FeedbackEmployeeViewScene);
    mainStage.setTitle("האם אתה ממליץ");
    mainStage.centerOnScreen();
    mainStage.show();

}

}

```

Navigators Interface:

```

Navigators Interface:
package com.example.hrpulse.Services.Interfaces;

import javafx.event.ActionEvent;
import javafx.fxml.FXMLLoader;
import javafx.scene.Node;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.stage.Stage;

import java.io.IOException;
import java.util.Objects;

public interface Navigators {

    default void navigateToLoginPage(ActionEvent event) throws IOException
{

    Parent loginPageViewParent =
FXMLLoader.load(Objects.requireNonNull(getClass().getResource("/com/example
/hrpulse/login-view.fxml")));
    Scene loginPageViewScene = new Scene(loginPageViewParent);
    Stage mainStage = (Stage) ((Node)
event.getSource()).getScene().getWindow();
    mainStage.setScene(loginPageViewScene);
    mainStage.setTitle("Hr-Pulse");
    mainStage.setResizable(false);
    mainStage.centerOnScreen();
    mainStage.show();
}

    default void navigateToMainPage(ActionEvent event) throws IOException {
    Parent managerPageViewParent =
FXMLLoader.load(Objects.requireNonNull(getClass().getResource("/com/example

```

```

    /hrpulse/UsersView/UsersView/Users_view.fxml"));
    Scene managerPageViewScene = new Scene(managerPageViewParent);
    Stage mainStage = (Stage) ((Node)
event.getSource().getScene().getWindow());
    mainStage.setScene(managerPageViewScene);
    mainStage.setTitle(" ראיית כניסה ");
    mainStage.centerOnScreen();
    mainStage.show();
}

// -----Department navigators -----
-----
    default void navigateToManageDepartment(ActionEvent event) throws
IOException {
    Parent manageDepartmentPageViewParent =
FXMLLoader.load(Objects.requireNonNull(getClass().getResource("/com/example
/hrpulse/DepartmentView/manageDepartment_view.fxml")));
    Scene manageDepartmentPageViewScene = new
Scene(manageDepartmentPageViewParent);
    Stage mainStage = (Stage) ((Node)
event.getSource().getScene().getWindow());
    mainStage.setScene(manageDepartmentPageViewScene);
    mainStage.setTitle(" ניהול מחלקות ");
    mainStage.centerOnScreen();
    mainStage.show();
}

    default void navigateToCreateDepartment(ActionEvent event) throws
IOException {
    // Load the FXML file and create the Scene as before
    Parent createDepartmentPageViewParent =
FXMLLoader.load(Objects.requireNonNull(getClass().getResource("/com/example
/hrpulse/DepartmentView/createNewDepartment_view.fxml")));
    Scene createDepartmentPageViewScene = new
Scene(createDepartmentPageViewParent);
    // Get the main Stage and set its properties
    Stage mainStage = (Stage) ((Node)
event.getSource().getScene().getWindow());
    mainStage.setScene(createDepartmentPageViewScene);
    mainStage.centerOnScreen();
    mainStage.setTitle(" מחלקה חדשה יצרת ");
    mainStage.show();
}

    default void navigateEditDepartment(ActionEvent event) throws
IOException {
    // Load the FXML file and create the Scene as before
    Parent editDepartmentPageViewParent =
FXMLLoader.load(Objects.requireNonNull(getClass().getResource("/com/example
/hrpulse/DepartmentView/editDepartment_view.fxml")));
    Scene editDepartmentPageViewScene = new
Scene(editDepartmentPageViewParent);
    // Get the main Stage and set its properties
    Stage mainStage = (Stage) ((Node)
event.getSource().getScene().getWindow());
    mainStage.setScene(editDepartmentPageViewScene);
    // Set the Stage properties for resizable and on Center of the
screen
    mainStage.centerOnScreen();
    mainStage.setTitle(" מחלקה נושא עדכון ");
    mainStage.show();
}

```

```

        }

        default void navigateToReportOfDepartment(ActionEvent event) throws
IOException {
    // Load the FXML file and create the Scene as before
    Parent ReportOfDepartmentPageViewParent =
FXMLLoader.load(Objects.requireNonNull(getClass().getResource("/com/example
/hrpulse/DepartmentView/reportDepartment_view.fxml")));
    Scene ReportOfDepartmentPageViewScene = new
Scene(ReportOfDepartmentPageViewParent);
    // Get the main Stage and set its properties
    Stage mainStage = (Stage) ((Node)
event.getSource()).getScene().getWindow();
    mainStage.setScene(ReportOfDepartmentPageViewScene);
    mainStage.centerOnScreen();
    mainStage.setTitle("מזהה מילקוט");
    mainStage.show();
}
}

```

ReportsNavigator:

```

package com.example.hrpulse.Services.Interfaces;

import javafx.event.ActionEvent;
import javafx.fxml.FXMLLoader;
import javafx.scene.Node;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.stage.Stage;
import java.io.IOException;
import java.util.Objects;

public interface ReportsNavigators {

    // -----ProductionOfReports navigator -----
    default void navigateToProductionOfReportsPage(ActionEvent event)
throws IOException {
    Parent ReportPageViewParent =
FXMLLoader.load(Objects.requireNonNull(getClass().getResource("/com/example
/hrpulse/productionOfReports_view/productionOfReports/productionOfReports_v
iew.fxml")));
    Scene ReportPageViewScene = new Scene(ReportPageViewParent);
    Stage mainStage = (Stage) ((Node)
event.getSource()).getScene().getWindow();
    mainStage.setScene(ReportPageViewScene);
    mainStage.setTitle("טפסון יומיות");
    mainStage.centerOnScreen();
    mainStage.show();
}

    // -----MonthlyShift navigator -----
    default void navigateToMonthlyShiftEmployee(ActionEvent event) throws
IOException {
    Parent MonthlyShiftEmployeeViewParent =
FXMLLoader.load(Objects.requireNonNull(getClass().getResource("/com/example
/hrpulse/productionOfReports_view/productionOfReports/MonthlyShiftEmployee_

```

```

report.fxml"));
    Scene MonthlyShiftEmployeeViewScene = new
Scene(MonthlyShiftEmployeeViewParent);
    Stage mainStage = (Stage) ((Node)
event.getSource()).getScene().getWindow();
    mainStage.setScene(MonthlyShiftEmployeeViewScene);
    mainStage.setTitle("התקנת תינוקות");
    mainStage.centerOnScreen();
    mainStage.show();
}
}

```

Now for the Controllers: Department controller code:

```

package com.example.hrpulse.Controllers.DepartmentController;

import com.example.hrpulse.Services.Database.DatabaseManager;
import com.example.hrpulse.Services.Interfaces.Navigators;
import com.example.hrpulse.Services.Objects.Department;
import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.scene.control.Alert;
import javafx.scene.control.TextField;
import org.hibernate.SessionFactory;

import java.io.IOException;

/**
 * The `CreateNewDepartmentController` class handles the logic for creating
a new department.
 */
public class CreateNewDepartmentController implements Navigators {

    // Constructors to allow for different ways of initializing the
controller
    private DatabaseManager databaseManager;
    private SessionFactory sessionFactory;

    public CreateNewDepartmentController() {

    }

    public CreateNewDepartmentController(DatabaseManager databaseManager) {
        this.databaseManager = databaseManager;
        this.sessionFactory = null;
    }

    public CreateNewDepartmentController(DatabaseManager databaseManager,
SessionFactory sessionFactory) {
        this.databaseManager = databaseManager;
        this.sessionFactory = sessionFactory;
    }
}

```

```

@FXML
private TextField tf_departmentName;

@FXML
private TextField tf_description;

/**
 * Handles the back button click event, navigating back to the manage
department page.
 */
@FXML
void backToManageDepartmentClick(ActionEvent event) throws IOException
{
    navigateToManageDepartment(event);
}

/**
 * Handles the save button click event, saving the new department to
the database.
 */
@FXML
void saveDepartmentClicked(ActionEvent event) {
    String departmentName = tf_departmentName.getText();
    String description = tf_description.getText();

    // Create a new Department instance with the entered details
    Department department = new Department();
    department.setDepartmentName(departmentName);
    department.setDescription(description);

    // Perform database operations to save the department
    databaseManager.performDatabaseOperations(department);

    // Show a confirmation dialog
    showConfirmationDialog("המחלקה ננתונה בהצלחה .");
}

/**
 * Shows an information dialog with the given message.
 *
 * @param message The message to be displayed in the dialog.
 */
private void showConfirmationDialog(String message) {
    Alert alert = new Alert(Alert.AlertType.INFORMATION);
    alert.setTitle("Department Saved");
    alert.setHeaderText(null);
    alert.setContentText(message);
    alert.showAndWait();
}
}

package com.example.hrpulse.Controllers.DepartmentController;

import com.example.hrpulse.Services.Database.DatabaseManager;
import com.example.hrpulse.Services.Interfaces.Navigators;
import com.example.hrpulse.Services.Objects.Department;
import javafx.event.ActionEvent;
import javafx.fxml.FXML;

```

```
import javafx.scene.control.*;
import org.hibernate.SessionFactory;

import java.io.IOException;
import java.util.ArrayList;
import java.util.List;

import static
com.example.hrpulse.Services.DatabaseManager.retrieveDepartments;

/**
 * The `EditDepartmentController` class handles the logic for editing and
removing departments.
 */
public class EditDepartmentController implements Navigators {
    private DatabaseManager databaseManager;
    private SessionFactory sessionFactory;

    public EditDepartmentController() {
    }

    public EditDepartmentController(DatabaseManager databaseManager) {
        this.databaseManager = databaseManager;
        this.sessionFactory = null;
    }

    public EditDepartmentController(DatabaseManager databaseManager,
SessionFactory sessionFactory) {
        this.databaseManager = databaseManager;
        this.sessionFactory = sessionFactory;
    }

    @FXML
    private Button btn_editButton;

    @FXML
    private Button btn_removeButton;

    @FXML
    private ChoiceBox<String> cb_departmentShow;

    @FXML
    private Label label_departmentDescription;

    @FXML
    private Label label_departmentName;

    @FXML
    private TextField tf_department;

    @FXML
    private TextField tf_departmentDescription;

    private List<Department> departments; // the list of departments

    /**
     * Initializes the controller.
     */
    @FXML
    public void initialize() {
        departments = retrieveDepartments();
```

```

        List<String> departmentNames =
retrieveDepartmentNames(departments);
        cb_departmentShow.getItems().addAll(departmentNames);
        cb_departmentShow.setDisable(false);

        // Initialize the choice box selection listener

cb_departmentShow.getSelectionModel().selectedItemProperty().addListener((o
bservable, oldValue, newValue) -> {
    if (newValue != null) {
        showFields();

        // Fetch the selected department by name
        Department selectedDepartment =
retrieveDepartmentByName(newValue);

        if (selectedDepartment != null) {

tf_department.setText(selectedDepartment.getDepartmentName());

tf_departmentDescription.setText(selectedDepartment.getDescription());
    }
    else {
        hideFields();
    }
}) ;

hideFields();
}

// Define the retrieveDepartmentByName method
private Department retrieveDepartmentByName(String departmentName) {
    for (Department department : departments) {
        if (department.getDepartmentName().equals(departmentName)) {
            return department;
        }
    }
    return null;
}

// Other methods for edit and remove actions can be implemented here

private void showFields() {
    tf_department.setVisible(true);
    label_departmentName.setVisible(true);
    tf_departmentDescription.setVisible(true);
    label_departmentDescription.setVisible(true);
    btn_editButton.setVisible(true);
    btn_removeButton.setVisible(true);
}

private void hideFields() {
    tf_department.setVisible(false);
    label_departmentName.setVisible(false);
    tf_departmentDescription.setVisible(false);
    label_departmentDescription.setVisible(false);
    btn_editButton.setVisible(false);
    btn_removeButton.setVisible(false);
}

```

```

    /**
     * Handles the back button click event, navigating back to the manage
     department page.
     */
    @FXML
    void backButtonClicked(ActionEvent event) throws IOException {
        navigateToManageDepartment(event);
    }

    /**
     * Handles the edit button click event, updating the selected
     department.
     */
    @FXML
    void editButtonClicked(ActionEvent event) {
        String selectedDepartmentName = cb_departmentShow.getValue();

        if (selectedDepartmentName != null) {
            Department selectedDepartment =
            retrieveDepartmentByName(selectedDepartmentName);

            if (selectedDepartment != null) {
                // Update the department entity with new data

                selectedDepartment.setDepartmentName(tf_department.getText());
                selectedDepartment.setDescription(tf_departmentDescription.getText());

                // Call the method in HR_Pulse to update the department

                com.example.hrpulse.Services.Database.DatabaseManager.performDatabaseOperations(selectedDepartment, true);

                // Display confirmation
                showConfirmationDialog("Updated successfully!", "המחלקה" +
                " בהצלחה נודכנה!");
            } else {
                // Display error
                showErrorDialog("Error", "המחלקה בטיפול שגיאה!");
            }
        }
    }

    /**
     * Handles the remove button click event, removing the selected
     department.
     */
    @FXML
    void removeButtonClicked(ActionEvent event) {
        String selectedDepartmentName = cb_departmentShow.getValue();

        if (selectedDepartmentName != null) {
            Department selectedDepartment =
            retrieveDepartmentByName(selectedDepartmentName);

            if (selectedDepartment != null) {
                // Call the method in HR_Pulse to remove the department

                com.example.hrpulse.Services.Database.DatabaseManager.performDatabaseOperations(selectedDepartment, false);
            }
        }
    }
}

```

```

        // Display confirmation
        showConfirmationDialog("Removed successfully!", "המזהה נסגרה");
    } else {
        // Display error
        showErrorDialog("Error", "המזהה במחיקת שגיאה!");
    }
}

// Define the retrieveDepartmentNames method
public static List<String> retrieveDepartmentNames(List<Department>
departments) {
    List<String> departmentNames = new ArrayList<>();

    for (Department department : departments) {
        departmentNames.add(department.getDepartmentName());
    }

    return departmentNames;
}

// Helper method to show a confirmation dialog
private void showConfirmationDialog(String title, String message) {
    Alert alert = new Alert(Alert.AlertType.INFORMATION);
    alert.setTitle(title);
    alert.setHeaderText(null);
    alert.setContentText(message);
    alert.showAndWait();
}

// Helper method to show an error dialog
private void showErrorDialog(String title, String message) {
    Alert alert = new Alert(Alert.AlertType.ERROR);
    alert.setTitle(title);
    alert.setHeaderText(null);
    alert.setContentText(message);
    alert.showAndWait();
}
}
}

```

```

package com.example.hrpulse.Controllers.DepartmentController;

import com.example.hrpulse.Services.Interfaces.Navigators;
import javafx.event.ActionEvent;
import javafx.fxml.FXML;

import java.io.IOException;

/**
 * The `ManageDepartmentController` class handles navigation related to
 * department management.
 */
public class ManageDepartmentController implements Navigators {

    /**
     * Navigates to the edit department page.
     */

```

```

        * @param event The ActionEvent triggered by the user.
        * @throws IOException If an I/O error occurs.
    */
@FXML
void navigateToEditDepartment(ActionEvent event) throws IOException {
    navigateEditDepartment(event);
}

/**
 * Navigates to the create department page.
 *
 * @param event The ActionEvent triggered by the user.
 * @throws IOException If an I/O error occurs.
 */
@FXML
void createDepartmentClicked(ActionEvent event) throws IOException {
    navigateToCreateDepartment(event);
}

/**
 * Navigates to the department details page (report of department).
 *
 * @param event The ActionEvent triggered by the user.
 * @throws IOException If an I/O error occurs.
 */
@FXML
void navigateToDepartmentDetails(ActionEvent event) throws IOException {
    navigateToReportOfDepartment(event);
}

/**
 * Navigates back to the main page.
 *
 * @param event The ActionEvent triggered by the user.
 * @throws IOException If an I/O error occurs.
 */
@FXML
void BackToMainClicked(ActionEvent event) throws IOException {
    navigateTo MainPage(event);
}

package com.example.hrpulse.Controllers.DepartmentController;

import com.example.hrpulse.Services.Database.DatabaseManager;
import com.example.hrpulse.Services.Interfaces.Navigators;
import com.example.hrpulse.Services.Objects.Company;
import com.example.hrpulse.Services.Objects.Department;
import com.example.hrpulse.Services.Objects.Employee;
import javafx.collections.FXCollections;
import javafx.collections.ObservableList;
import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.scene.control.Button;
import javafx.scene.control.ListView;
import org.hibernate.SessionFactory;

```

```
import java.io.IOException;
import java.util.List;

import static
com.example.hrpulse.Services.Database.DatabaseManager.retrieveDepartments;
import static
com.example.hrpulse.Services.Database.DatabaseManager.retrieveEmployees;

/**
 * The `ReportDepartmentController` class manages the UI for reporting
department details.
 */
public class ReportDepartmentController implements Navigators {

    private DatabaseManager databaseManager;
    private SessionFactory sessionFactory;

    public ReportDepartmentController() {
    }

    public ReportDepartmentController(DatabaseManager databaseManager) {
        this.databaseManager = databaseManager;
        this.sessionFactory = null;
    }

    public ReportDepartmentController(DatabaseManager databaseManager,
SessionFactory sessionFactory) {
        this.databaseManager = databaseManager;
        this.sessionFactory = sessionFactory;
    }

    @FXML
    private ListView<Department> lv_departments;

    private ObservableList<Department> departmentList =
FXCollections.observableArrayList();

    /**
     * Initializes the controller. Populates the department list in the UI.
     */
    public void initialize() {
        List<Employee> employees = retrieveEmployees();
        List<Department> departments = retrieveDepartments();
        departmentList.addAll(departments);
        lv_departments.setItems(departmentList);

        Company myCompany = Company.getMyCompany();
        if (myCompany != null) {
            // Add departments to the company and display them
            for (Department department : departments) {
                myCompany.addDepartment(department);
                System.out.println(department);
            }
        }

        // Add employees to the company and assign them to their
respective departments
        for (Employee employee : employees) {
            myCompany.addEmployee(employee);
            System.out.println(employee);
            String departmentName = employee.getDepartment();
            Department matchingDepartment =

```

```

myCompany.getDepartmentByName(departmentName);
        if (matchingDepartment != null) {
            // If a matching department is found, add the employee
            to it
                matchingDepartment.addEmployee(employee);
            } else {
                System.err.println("+" : לטעוב מתאימה מחלקה נמצאה לא(" +
employee.getFirstName());
            }
        }
    } else {
        System.err.println("+" . некоּ מוגדרת לא המחלקה(");
    }
}

/**
 * Handles the action when the "Back to Manage Department" button is
clicked.
 *
 * @param event The ActionEvent triggered by the user.
 * @throws IOException If an I/O error occurs.
 */
@FXML
void backToManageDepartment(ActionEvent event) throws IOException {
    navigateToManageDepartment(event);
}
}

```

Employee Controller:

```

package com.example.hrpulse.Controllers.EmployeeController;

import com.example.hrpulse.Services.Database.DatabaseManager;
import com.example.hrpulse.Services.Interfaces.EmployeeNavigators;
import com.example.hrpulse.Services.Objects.Department;
import com.example.hrpulse.Services.Objects.Employee;
import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.scene.control.*;
import org.hibernate.SessionFactory;
import java.io.IOException;
import java.time.LocalDate;
import java.time.ZoneId;
import java.util.Date;
import java.util.List;
import static
com.example.hrpulse.Controllers.DepartmentController.EditDepartmentController.retrieveDepartmentNames;
import static
com.example.hrpulse.Services.Database.DatabaseManager.retrieveDepartments;
import static
com.example.hrpulse.Services.Database.DatabaseManager.retrieveEmployees;

/**
 * The CreateEmployeePageController handles the logic for creating a new
employee.

```

```
 */
public class CreateEmployeePageController implements EmployeeNavigators {

    // Constructors to allow for different ways of initializing the
    controller
    private DatabaseManager databaseManager;
    private SessionFactory sessionFactory;

    public CreateEmployeePageController() {
        // Default constructor
    }

    public CreateEmployeePageController(DatabaseManager databaseManager) {
        this.databaseManager = databaseManager;
        // Initialize the controller here if needed
        this.sessionFactory = null; // Or initialize it if needed
    }

    public CreateEmployeePageController(DatabaseManager databaseManager,
SessionFactory sessionFactory) {
        this.databaseManager = databaseManager;
        this.sessionFactory = sessionFactory;
    }

    // FXML elements injected from the corresponding FXML file
    @FXML
    private TextField tf(firstName);
    @FXML
    private TextField tf(lastName);
    @FXML
    private TextField tf(employeeID);
    @FXML
    private TextField tf(email);
    @FXML
    private TextField tf(phoneNumber);
    @FXML
    private ChoiceBox<String> cb(gender);
    @FXML
    private ChoiceBox<String> cb(role);
    @FXML
    private TextField tf(password);
    @FXML
    private ChoiceBox<String> cb(department);
    @FXML
    private DatePicker dp(dateOfBirth);
    @FXML
    private DatePicker dp(startDate);
    @FXML
    private CheckBox cb(isHourly);
    @FXML
    private CheckBox cb(isPerMoth);
    @FXML
    private TextField tf(salaryPerHour);
    @FXML
    private TextField tf(perMonth);
    @FXML
    private TextField tf(salaryToTravel);
    @FXML
    private TextField tf(bankNumber);
    @FXML
    private TextField tf(accountNumber);
```

```

@FXML
private TextField tf_sneefBankCode;

@FXML
void initialize() {
    // Initialization logic for the controller

    // Add a change listener to cb_isPerMoth
    cb_isPerMoth.selectedProperty().addListener((observable, oldValue,
newValue) -> {
        // Disable or enable cb_isHourly and tf_salaryPerHour based on
newValue
        cb_isHourly.setDisable(newValue);
        tf_salaryPerHour.setDisable(newValue);
    });
    cb_isHourly.selectedProperty().addListener((observable, oldValue,
newValue) -> {
        // Disable or enable cb_isHourly and tf_salaryPerHour based on
newValue
        cb_isPerMoth.setDisable(newValue);
        tf_perMonth.setDisable(newValue);
    });
}

cb_role.getSelectionModel().selectedItemProperty().addListener((observable,
oldValue, newValue) -> {
    // Check if the selected role is "secretariat" or
"headOfDepartment"
    if ("secretary".equals(newValue) ||
"headOfDepartment".equals(newValue)) {
        // Enable and show the disabled label
        tf_password.setDisable(false);
        tf_password.setVisible(true);
    } else {
        // Otherwise, disable and hide the label
        tf_password.setDisable(true);
        tf_password.setVisible(false);
    }
});
// declare a list of departments
List<Department> departments;
// retrieve the department from database by using the static method
from the hr_pulse class
departments = retrieveDepartments();

// Check if departments are not null before populating the ComboBox
if (departments != null) {
    // Retrieve department names using the static method that is
presumably in the EditDepartment class
    List<String> departmentNames =
retrieveDepartmentNames(departments);

    // Populate the ComboBox with department names
    cb_department.getItems().addAll(departmentNames);
}
}

@FXML
void goToMain(ActionEvent event) throws IOException {
    navigateToManageEmployeePage(event);
}

```

```

}

@FXML
void saveButtonClicked(ActionEvent event) {
    // Logic for saving employee data
    // Collect data from input fields
    // Validate first name
    String firstName = tf(firstName.getText().trim());
    if (firstName.isEmpty()) {
        showErrorDialog("השם הראשי נא");
        return;
    }
    if (firstName.length() < 2) {
        showErrorDialog("השם הראשי קצר");
        return;
    }

    // Validate last name
    String lastName = tf(lastName.getText().trim());
    if (lastName.isEmpty()) {
        showErrorDialog("השם האחרון נא");
        return;
    }
    if (lastName.length() < 2) {
        showErrorDialog("השם האחרון קצר");
        return;
    }

    // Validate employee ID
    String employeeIDText = tf(employeeID.getText().trim());
    if (employeeIDText.isEmpty()) {
        showErrorDialog("המספרן דרכן נא");
        return;
    }
    int employeeID = 0;

    try {
        employeeID = Integer.parseInt(employeeIDText);
        if (employeeIDText.trim().length() != 9) {
            showErrorDialog("מספרן דרכן חייבת להיות 9");
            return;
        }
    } catch (NumberFormatException e) {
        showErrorDialog("מספרן דרכן חייבת להיות 9");
        return;
    }

    // Check if the employee ID is unique
    if (!isEmployeeIDUnique(employeeID)) {
        showErrorDialog("מספרן דרכן כבר קיים במערכת");
        return;
    }
    String email = tf(email.getText());
    if (email.isEmpty()) {
        showErrorDialog("הכתובת האלקטרונית דואר הכנס נא");
        return;
    }

    // Validate phone number
    String phoneNumber = tf(phoneNumber.getText().trim());

```

```

if (phoneNumber.isEmpty()) {
    showErrorDialog("אנו צריכים מספר טלפון נייד.");
    return;
}
String department = cb_department.getValue();
// Check if a department name is selected
if (department == null || department.isEmpty()) {
    showErrorDialog("אנו צריכים שם מחלקת.");
    return; // Exit the method
}
String gender = cb_gender.getValue();
// Validate role
String role = cb_role.getValue();
if (role == null || role.isEmpty()) {
    showErrorDialog("אנו צריכים תפקיד.");
    return;
}
// If role requires a password, validate the password field
if ("secretary".equals(role) || "headOfDepartment".equals(role)) {
    String password = tf_password.getText().trim();
    if (password.isEmpty()) {
        showErrorDialog("זה לא עובד סיסמה בחר איזה!");
        return;
    }
}
String password = tf_password.getText();

// Retrieve the selected date from the DatePicker
LocalDate dateOfBirth = dp_dateOfBirth.getValue();

LocalDate employmentStartDate = dp_startDate.getValue();

boolean isHourly = cb_isHourly.isSelected();
//-----
String salaryPerHourText = tf_salaryPerHour.getText();
int salaryPerHour = 0;
try {
    salaryPerHour = Integer.parseInt(salaryPerHourText);
} catch (NumberFormatException e) {

}

boolean isPerMonth = cb_isPerMonth.isSelected();
//-----
String salaryPerMonthText = tf_perMonth.getText();
Double salaryPerMonth = 0.0;
try {
    salaryPerMonth = Double.parseDouble(salaryPerMonthText);
} catch (NumberFormatException e) {

}
//-----
String salaryToTravelText = tf_salaryToTravel.getText();
Double salaryToTravel = 0.0;
try {
    salaryToTravel = Double.parseDouble(salaryToTravelText);
} catch (NumberFormatException e) {
}

```

```

        }

-----
```

```

        String bankNumberText = tf_bankNumber.getText();
        int bankNumber = 0;
        try {
            bankNumber = Integer.parseInt(bankNumberText);
        } catch (NumberFormatException e) {

        }
-----
```

```

        String accountNumber = tf_accountNumber.getText();
        //-----validate that sneef has just a numbers
        String sneefBankCodeText = tf_sneefBankCode.getText();
        int sneefBankCode = 0;
        try {
            sneefBankCode = Integer.parseInt(sneefBankCodeText);
        } catch (NumberFormatException e) {

        }
-----
```

```

        // Create an Employee object
        Employee employee = new Employee();
        employee.setFirstName(firstName);
        employee.setLastName(lastName);
        employee.setEmployeeId(employeeID);
        employee.setPhoneNumber(phoneNumber);
        employee.setEmail(email);
        employee.setGender(gender);
        employee.setEmployeeRole(role);
        employee.setDateOfBirth(dateOfBirth);
        employee.setPassword(password);
        employee.setHourly(isHourly);
        employee.setHourlyRate(salaryPerHour);
        employee.setSalaryPerMonth(salaryPerMonth);
        employee.setSalaryToTravel(salaryToTravel);
        employee.getBankInfo().setBankNumber(bankNumber);
        employee.getBankInfo().setAccountNumber(accountNumber);
        employee.getBankInfo().setBankSneefCode(sneefBankCode);
        if (employmentStartDate != null) {

employee.setEmploymentStartDate(Date.from(employmentStartDate.atStartOfDay(
ZoneId.systemDefault()).toInstant()));
        } else {
            // Set the employment start date to the current day
            LocalDate currentDay = LocalDate.now();

employee.setEmploymentStartDate(Date.from(currentDay.atStartOfDay(ZoneId.sy
stemDefault()).toInstant()));
        }

        String departmentName = cb_department.getValue(); // Get the
selected department name
    }
}

```

```

        // Retrieve the selected department based on its name
        Department selectedDepartment =
getDepartmentByName(departmentName);

        // Associate the employee with the selected department
employee.setDepartment(selectedDepartment.getDepartmentName());
// selectedDepartment.addEmployee(employee);
        // Save the data to the database using HR_Pulse's method

        databaseManager.performDatabaseOperations(employee);

        // Optionally, display a confirmation message
showConfirmationDialog("הצלחה התווסף העובד נטעני");
clearInputFields();

}

// Method to retrieve a department by its name
private Department getDepartmentByName(String departmentName) {
    // declare a list of departments
List<Department> allDepartments;
    // retrieve the department from database by using the static method
from the hr_pulse class
    allDepartments = retrieveDepartments();
    for (Department department : allDepartments) {
        if (department.getDepartmentName().equals(departmentName)) {
            return department;
        }
    }
    return null; // Department not found
}

// Helper method to show a confirmation dialog
private void showConfirmationDialog(String message) {
    Alert alert = new Alert(Alert.AlertType.INFORMATION);
    alert.setTitle(" Confirm ");
    alert.setHeaderText(null);
    alert.setContentText(message);
    alert.showAndWait();
}

// Helper method to show an error dialog
private void showErrorDialog(String message) {
    Alert alert = new Alert(Alert.AlertType.ERROR);
    alert.setTitle(" Error ");
    alert.setHeaderText(null);
    alert.setContentText(message);
    alert.showAndWait();
}

// Method to check if an employee ID is unique
private boolean isEmployeeIDUnique(int employeeID) {

    List<Employee> employees;

    employees=retrieveEmployees();
    for (Employee employee: employees
    ) {

```

```

        if(employee.getEmployeeId()==employeeID) return false;
    }
    return true; // Replace with your actual logic
}

// Method to clear input fields after saving
private void clearInputFields() {
    tf(firstName).clear();
    tf(lastName).clear();
    tf(employeeID).clear();
    tf(email).clear();
    tf(phoneNumber).clear();
    cb(gender).getSelectionModel().clearSelection();
    cb(role).getSelectionModel().clearSelection();
    tf(password).clear();
    cb(department).getSelectionModel().clearSelection();
    dp(dateOfBirth).setValue(null);
    dp(startDate).setValue(null);
    cb(isHourly).setSelected(false);
    tf(salaryPerHour).clear();
    cb(isPerMoth).setSelected(false);
    tf(perMonth).clear();
    tf(salaryToTravel).clear();
    tf(bankNumber).clear();
    tf(accountNumber).clear();
    tf(sneefBankCode).clear();
}
}

```

```

package com.example.hrpulse.Controllers.EmployeeController;

import com.example.hrpulse.Services.Database.DatabaseManager;
import com.example.hrpulse.Services.Interfaces.EmployeeNavigators;
import com.example.hrpulse.Services.Objects.Employee;
import com.example.hrpulse.HR_Pulse;
import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.scene.control.*;
import javafx.scene.layout.GridPane;
import org.hibernate.SessionFactory;
import java.io.IOException;
import java.util.ArrayList;
import java.util.List;

import static com.example.hrpulse.Services.Database.DatabaseManager.retrieveEmployees;

/**
 * The EditEmployeeController handles the logic for editing employee details.
 */
public class EditEmployeeController implements EmployeeNavigators {

    // Constructors to allow for different ways of initializing the

```

```
controller
    private DatabaseManager databaseManager;

    private SessionFactory sessionFactory;

    public EditEmployeeController() {
    }

    public EditEmployeeController(DatabaseManager databaseManager) {
        this.databaseManager = databaseManager;
        this.sessionFactory = null;
    }

    public EditEmployeeController(DatabaseManager databaseManager,
SessionFactory sessionFactory) {
        this.databaseManager = databaseManager;
        this.sessionFactory = sessionFactory;
    }

    @FXML
    private TextField tf(firstName);
    @FXML
    private TextField tf(lastName);
    @FXML
    private TextField tf(employeeID);
    @FXML
    private TextField tf(email);
    @FXML
    private TextField tf(phoneNumber);
    @FXML
    private ChoiceBox<String> cb(gender);
    @FXML
    private ChoiceBox<String> cb(role);
    @FXML
    private TextField tf(password);

    @FXML
    private CheckBox cb(isHourly);
    @FXML
    private TextField tf(salaryPerHour);
    @FXML
    private CheckBox cb(isPerMoth);
    @FXML
    private TextField tf(perMonth);
    @FXML
    private TextField tf(salaryToTravel);
    @FXML
    private TextField tf(bankNumber);
    @FXML
    private TextField tf(accountNumber);
    @FXML
    private TextField tf(sneefBankCode);
    @FXML
    private ChoiceBox<String> cb(retrieveEmployee);
    @FXML
    private GridPane gp(editEmployee);
    private List<Employee> employeeList;

    @FXML
    public void initialize() {
        // Initialization logic for the controller
    }
}
```

```

employeeList = retrieveEmployees();
List<String> employeeFullName =
retrieveEmployeesNames(employeeList);
cb_retrieveEmployee.getItems().addAll(employeeFullName);

// Set an event handler for the cb_retrieveEmployee ChoiceBox
gp_editEmployee.setVisible(false);

cb_retrieveEmployee.getSelectionModel().selectedItemProperty().addListener((observable, oldValue, newValue) -> {
    if (newValue != null) {
        gp_editEmployee.setVisible(true);
        fillEmployeeDetails(newValue);
    } else {
        gp_editEmployee.setVisible(false);
    }
});

// Add a change listener to cb_isPerMoth
cb_isPerMoth.selectedProperty().addListener((observable, oldValue, newValue) -> {
    // Disable or enable cb_isHourly and tf_salaryPerHour based on
newValue
    cb_isHourly.setDisable(newValue);
    tf_salaryPerHour.setDisable(newValue);
});
cb_isHourly.selectedProperty().addListener((observable, oldValue, newValue) -> {
    // Disable or enable cb_isHourly and tf_salaryPerHour based on
newValue
    cb_isPerMoth.setDisable(newValue);
    tf_perMonth.setDisable(newValue);
});

cb_role.getSelectionModel().selectedItemProperty().addListener((observable,
oldValue, newValue) -> {
    // Check if the selected role is "secretariat" or
"headOfDepartment"
    if ("secretary".equals(newValue) ||
"headOfDepartment".equals(newValue)) {
        // Enable and show the disabled label
        tf_password.setDisable(false);
        tf_password.setVisible(true);
    } else {
        // Otherwise, disable and hide the label
        tf_password.setDisable(true);
        tf_password.setVisible(false);
    }
});
}

// Method to fill the employee details in the form
private void fillEmployeeDetails(String fullName) {
    // Extract the first name and last name from the full name
    String[] names = fullName.split(" ");
    String firstName = names[0];
    String lastName = names[1];
    // Find the employee with the given first name and last name
    Employee selectedEmployee = findEmployeeByName(firstName,
lastName);
}

```

```

// Fill the fields with the details of the selected employee
tf(firstName).setText(selectedEmployee.getFirstName());
tf(lastName).setText(selectedEmployee.getLastName());

tf_employeeID.setText(String.valueOf(selectedEmployee.getEmployeeId()));
tf_email.setText(selectedEmployee.getEmail());
tf_phoneNumber.setText(selectedEmployee.getPhoneNumber());
cb_gender.setValue(selectedEmployee.getGender());
cb_role.setValue(selectedEmployee.getEmployeeRole());
cb_isHourly.setSelected(selectedEmployee.isHourly());
tf_password.setText(selectedEmployee.getPassword());

tf_salaryPerHour.setText(String.valueOf(selectedEmployee.getHourlyRate()));
tf_perMonth.setText(String.valueOf(selectedEmployee.getSalaryPerMonth()));

tf_salaryToTravel1.setText(String.valueOf(selectedEmployee.getSalaryToTravel()));

tf_bankNumber.setText(String.valueOf(selectedEmployee.getBankInfo().getBankNumber()));

tf_acountNumber.setText(selectedEmployee.getBankInfo().getAccountNumber());

tf_sneefBankCode.setText(String.valueOf(selectedEmployee.getBankInfo().getBankSneefCode()));
}

// Method to find an employee by first name and last name
private Employee findEmployeeByName(String firstName, String lastName)
{
    for (Employee employee : employeeList) {
        if (employee.getFirstName().equals(firstName) &&
employee.getLastName().equals(lastName)) {
            return employee;
        }
    }
    return null; // Employee not found
}

// Method to retrieve employee names for the choice box
private List<String> retrieveEmployeesNames(List<Employee>
employeeList) {
    List<String> employeeName = new ArrayList<>();
    for (Employee employee : employeeList
    ) {
        employeeName.add(employee.getFirstName() + " " +
employee.getLastName());
    }
    return employeeName;
}

// Method to handle employee deletion

public void deletebtnClick(ActionEvent actionEvent) {

    String fullName = cb_retrieveEmployee.getValue();
    if (fullName != null) {
        // Extract the first name and last name from the full name
        String[] names = fullName.split(" ");

```

```

        String firstName = names[0];
        String lastName = names[1];

        // Find the employee with the given first name and last name
        Employee selectedEmployee = findEmployeeByName(firstName,
lastName);

        databaseManager.employeePDO(selectedEmployee, false);

        // Clear the choice box and hide the grid pane
        cb_retrieveEmployee.setValue(null);
        gp_editEmployee.setVisible(false);
    }
}

// Method to handle saving updated employee details
public void saveBtnClicked(ActionEvent actionEvent) {
    String fullName = cb_retrieveEmployee.getValue();
    if (fullName != null) {
        // Extract the first name and last name from the full name
        String[] names = fullName.split(" ");
        String firstName = names[0];
        String lastName = names[1];

        // Find the employee with the given first name and last name
        Employee selectedEmployee = findEmployeeByName(firstName,
lastName);

        // Update the selected employee with the new details
        selectedEmployee.setEmail(tf_email.getText());
        selectedEmployee.setPhoneNumber(tf_phoneNumber.getText());
        selectedEmployee.setGender(cb_gender.getValue());
        selectedEmployee.setEmployeeRole(cb_role.getValue());
        selectedEmployee.setHourly(cb_isHourly.isSelected());
        selectedEmployee.setPassword(tf_password.getText());

        selectedEmployee.setHourlyRate(Double.parseDouble(tf_salaryPerHour.getText()));
        selectedEmployee.setSalaryPerMonth(Double.parseDouble(tf_perMonth.getText()));
        selectedEmployee.setSalaryToTravel(Double.parseDouble(tf_salaryToTravel.getText()));

        selectedEmployee.getBankInfo().setBankNumber(Integer.parseInt(tf_bankNumber.getText()));

        selectedEmployee.getBankInfo().setAccountNumber(tf_accountNumber.getText());
        selectedEmployee.getBankInfo().setBankSneefCode(Integer.parseInt(tf_sneefBankCode.getText()));

        databaseManager.employeePDO(selectedEmployee, true);

        // Clear the choice box and hide the grid pane
        cb_retrieveEmployee.setValue(null);
        gp_editEmployee.setVisible(false);
    }
}

```

```

// Method to navigate back to the manage employee page
public void backbtn(ActionEvent actionEvent) throws IOException {
    navigateToManageEmployeePage(actionEvent);
}

}

package com.example.hrpulse.Controllers.EmployeeController;

import com.example.hrpulse.Services.CSV.CsvRow;
import com.example.hrpulse.Services.CSV.UploadCsvFile;
import com.example.hrpulse.Services.Interfaces.EmployeeNavigators;
import com.example.hrpulse.Services.Objects.Employee;
import com.example.hrpulse.Session.UserSession;
import javafx.collections.FXCollections;
import javafx.collections.ObservableList;
import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.scene.control.*;
import javafx.scene.control.cell.PropertyValueFactory;
import javafx.scene.control.cell.TextFieldTableCell;
import com.example.hrpulse.Services.Database.DatabaseManager;
import org.hibernate.Session;
import org.hibernate.Transaction;
import org.hibernate.query.Query;
import java.io.IOException;
import java.util.*;

import static com.example.hrpulse.HR_Pulse.sessionFactory;

/**
 *
 * Controller class for Employee Shift Data.
 */
public class EditEmployeeShiftController implements EmployeeNavigators {

    @FXML
    private TableColumn<CsvRow, String> tc_totalWorkHrs;

    @FXML
    private TableColumn<CsvRow, String> tc_breakTime;

    @FXML
    private TableColumn<CsvRow, String> tc_exitHour;

    @FXML
    private TableColumn<CsvRow, String> tc_enterHour;

    @FXML
    private TableColumn<CsvRow, String> tc_date;

    @FXML
    private TableColumn<CsvRow, String> tc_employeeId;

    @FXML

```

```

TableColumn<CsvRow, Void> deleteColumn = new TableColumn<>("Delete");

@FXML
private TableView<CsvRow> tableViewCSVData;

private UploadCsvFile uploadCsvFile;

@FXML
private Button loadLastSavedDataButton;

@FXML
private TextField tf_employeeID;

@FXML
private Button saveButton;

private String csvFilePath;

// Keep track of edited rows and their IDs
private Map<String, CsvRow> editedRowsMap = new HashMap<>();

UserSession userSession = UserSession.getInstance();

/**
 * Initializes the EditEmployeeShiftController.
 */
public EditEmployeeShiftController() {

}

/**
 * Handles the upload CSV file action.
 *
 * @param event The ActionEvent triggered by the upload button.
 */
@FXML
void uploadCsvFile(ActionEvent event) {
    // Get the current user from UserSession
    Employee currentUser = userSession.getCurrentUser();

    // Check if the current user is null (handle it appropriately)
    if (currentUser == null) {
        // Handle the case where currentUser is null
        System.err.println("Error: currentUser is null in
uploadCsvFile");
        return;
    }

    uploadCsvFile.upload();

    // Set the CSV file path in the controller
    setCsvFilePath(uploadCsvFile.getFilePath());
}

/**
 * Initializes the controller and sets up event handlers.
 */

```

```

@FXML
public void initialize() {

    // Set action for the last saved button
    loadLastSavedDataButton.setOnAction(this::loadLastSavedDataIntoTableview);

    // Initialize UploadCsvFile with TableView and data type
    uploadCsvFile = new UploadCsvFile(tableViewCSVData,
"employeeshiftdata");

    // Setup column cell value factories and edit commit handlers

    tc_totalWorkHrs.setCellValueFactory(new
PropertyValueFactory<>("totalWorkHours"));

    tc_totalWorkHrs.setCellFactory(TextFieldTableCell.forTableColumn());
    tc_totalWorkHrs.setOnEditCommit(event -> {
        CsvRow editedRow = event.getRowValue();
        editedRow.setTotalWorkHours(event.getNewValue());
        event.getRowValue().setTotalWorkHours(event.getNewValue());
        try {
            uploadCsvFile.updateRowInCsv(event.getRowValue());
        } catch (IOException e) {
            throw new RuntimeException(e);
        }
    });
    tc_totalWorkHrs.setEditable(true);

    tc_breakTime.setCellValueFactory(new
PropertyValueFactory<>("breakTime"));
    tc_breakTime.setCellFactory(TextFieldTableCell.forTableColumn());
    tc_breakTime.setOnEditCommit(event -> {
        CsvRow editedRow = event.getRowValue();
        editedRow.setBreakTime(event.getNewValue());
        event.getRowValue().setBreakTime(event.getNewValue());
        try {
            uploadCsvFile.updateRowInCsv(event.getRowValue());
        } catch (IOException e) {
            throw new RuntimeException(e);
        }
    });
    tc_breakTime.setEditable(true);

    tc_exitHour.setCellValueFactory(new
PropertyValueFactory<>("exitHour"));
    tc_exitHour.setCellFactory(TextFieldTableCell.forTableColumn());
    tc_exitHour.setOnEditCommit(event -> {
        CsvRow editedRow = event.getRowValue();
        editedRow.setExitHour(event.getNewValue());
        event.getRowValue().setExitHour(event.getNewValue());
        try {
            uploadCsvFile.updateRowInCsv(event.getRowValue());
        } catch (IOException e) {
            throw new RuntimeException(e);
        }
    });
}

```

```

        }

    });

    tc_enterHour.setCellValueFactory(new
PropertyValueFactory<>("startHour"));
    tc_enterHour.setCellFactory(TextFieldTableCell.forTableColumn());
    tc_enterHour.setOnEditCommit(event -> {
        CsvRow editedRow = event.getRowValue();
        editedRow.setStartHour(event.getNewValue());
        event.getRowValue().setStartHour(event.getNewValue());
        try {
            uploadCsvFile.updateRowInCsv(event.getRowValue());
        } catch (IOException e) {
            throw new RuntimeException(e);
        }
    });

    tc_date.setCellValueFactory(new PropertyValueFactory<>("date"));
    tc_date.setCellFactory(TextFieldTableCell.forTableColumn());
    tc_date.setOnEditCommit(event -> {
        CsvRow editedRow = event.getRowValue();
        editedRow.setDate(event.getNewValue());
        try {
            uploadCsvFile.updateRowInCsv(event.getRowValue());
        } catch (IOException e) {
            throw new RuntimeException(e);
        }
    });

    tc_employeeId.setCellValueFactory(new
PropertyValueFactory<>("employeeId"));
    tc_employeeId.setCellFactory(TextFieldTableCell.forTableColumn());
    tc_employeeId.setOnEditCommit(event -> {
        CsvRow editedRow = event.getRowValue();
        editedRow.setEmployeeId(event.getNewValue());
        event.getRowValue().setEmployeeId(event.getNewValue());
        try {
            uploadCsvFile.updateRowInCsv(event.getRowValue());
        } catch (IOException e) {
            throw new RuntimeException(e);
        }
    });

    });

    // Setup delete column with a delete button
    deleteColumn.setCellFactory(param -> new TableCell<>() {
        private final Button deleteButton = new Button("הסרה");
        {

            // Set action for the delete button
            deleteButton.setOnAction(event -> {
                CsvRow selectedRow =
getTableView().getItems().get(getIndex());
                handleDeleteButton(selectedRow);
            });
        }
    });
}

```

```

@Override
protected void updateItem(Void item, boolean empty) {
    super.updateItem(item, empty);
    // Display the delete button if the row is not empty
    if (empty) {
        setGraphic(null);
    } else {
        setGraphic(deleteButton);
    }
}
});

// Add the delete column to the TableView
tableViewCSVData.getColumns().add(deleteColumn);

// Set edit commit handlers for each column
setEditCommitHandler(tc_breakTime);
setEditCommitHandler(tc_exitHour);
setEditCommitHandler(tc_enterHour);
setEditCommitHandler(tc_date);
setEditCommitHandler(tc_employeeId);

// Enable TableView editing
tableViewCSVData.setEditable(true);
}

/**
 * Sets up the edit commit handler for a TableColumn.
 *
 * @param column The TableColumn for which the edit commit handler is
 * set up.
 */
private void setEditCommitHandler(TableColumn<CsvRow, String> column) {
    column.setCellFactory(TextFieldTableCell.forTableColumn());
    column.setOnEditCommit(event -> handleEditCommit(event, column));
    column.setEditable(true);
}

/**
 * Handles the delete button action for a selected row.
 *
 * @param selectedRow The CsvRow selected for deletion.
 */
private void handleDeleteButton(CsvRow selectedRow) {
    // Display a confirmation dialog before deleting the row
    Alert alert = new Alert(Alert.AlertType.CONFIRMATION);
    alert.setTitle("האם מCONFIRMATION");
    alert.setHeaderText(null);
    alert.setContentText("האם שברצונך בטוח אתה שזאת שורה למחוק?");

    // Process the user's choice from the confirmation dialog
    Optional<ButtonType> result = alert.showAndWait();
    if (result.isPresent() && result.get() == ButtonType.OK) {
        try {
            // Remove the row from CSV and database, then update
            TableView
                uploadCsvFile.removeRowFromCsvAndDatabase(selectedRow);
                tableViewCSVData.getItems().remove(selectedRow);
                tableViewCSVData.getSelectionModel().clearSelection();
                showAlert("השורה נמחקה בהצלחה.");
        }
    }
}

```

```

        } catch (Exception e) {
            e.printStackTrace();
            showAlert("שורה למחיקת בנסיעות שגיאח");
        }
    }
}

/**
 * Handles the action when the add row button is clicked.
 *
 * @param event The ActionEvent triggered by the add row button.
 */
@FXML
private void addRowButtonClicked(ActionEvent event) {
    // Create a new CsvRow with default values
    CsvRow newRow = new CsvRow("", "", "", "", "", "", "");

    // Check for duplicate in the TableView
    if (isDuplicateRowInTableView(newRow)) {
        showAlert("הכטש ישן מזק או תאריך שונה אונא, קיימם כבר חזה התאריך");
        return;
    }

    // Mark the new row as externally added
    uploadCsvFile.markExternallyAddedRow(newRow);

    // Add the new row to the TableView
    tableViewCSVData.getItems().add(newRow);

    // Scroll to and select the new row
    tableViewCSVData.scrollTo(newRow);
    tableViewCSVData.getSelectionModel().select(newRow);
    tableViewCSVData.edit(tableViewCSVData.getItems().indexOf(newRow),
    tableViewCSVData.getColumns().get(0));

    // Track the new row in the map
    editedRowsMap.put(newRow.getCompositeKey(), newRow);
}

/**
 * Converts ObservableList of CsvRow to a List of String arrays for CSV
processing.
 *
 * @param csvRows The ObservableList of CsvRow to be converted.
 * @return List of String arrays representing the CSV data.
 */
private List<String[]> convertToStringCsv(ObservableList<CsvRow>
csvRows) {
    List<String[]> stringCsvData = new ArrayList<>();
    for (CsvRow row : csvRows) {
        stringCsvData.add(new String[]{
            row.getTotalWorkHours(),
            row.getBreakTime(),
            row.getExitHour(),
            row.getStartHour(),
            row.getDate(),
            row.getEmployeeId(),
        });
    }
    return stringCsvData;
}

```

```

}

/**
 * Sets the CSV file path in the controller.
 *
 * @param csvFilePath The path to the CSV file.
 */
public void setCsvFilePath(String csvFilePath) {
    this.csvFilePath = csvFilePath;
}

/**
 * Loads the last saved data from the database into the TableView.
 *
 * @param actionEvent The ActionEvent triggered by the load last saved
data button.
 */
@FXML
private void loadLastSavedDataIntoTableview(ActionEvent actionEvent) {
    if (!uploadCsvFile.isTableViewLoaded()) {
        // Load last saved data from the database
        List<CsvRow> lastSavedData =
DatabaseManager.loadLastSavedData("employeeshiftdata", CsvRow.class);
        if (lastSavedData.isEmpty()) {
            showAlert("נתונים מחסן להציג אחרוניים נתונים נמצאו לא");
        } else {
            // Check if the database is empty and save all data to it
            if (DatabaseManager.isEmpty("employeeshiftdata")) {
                saveAllDataToDatabase();
            } else {
                // Load the data into the TableView
                ObservableList<CsvRow> dataModels =
FXCollections.observableArrayList(lastSavedData);
                tableViewCSVData.setItems(dataModels);
            }
        }
    } else {
        showAlert("בשימוש כבר הטבלה כי אחרוניים נתונים להציג ניתן לא");
    }
}

/**
 * Displays an information alert with the given message.
 *
 * @param message The message to be displayed in the alert.
 */
private void showAlert(String message) {
    Alert alert = new Alert(Alert.AlertType.INFORMATION);
    alert.setTitle("Information");
    alert.setHeaderText(null);
    alert.setContentText(message);
    alert.showAndWait();
}

/**
 * Handles the save button click event.
 *
 * @param event The ActionEvent triggered by the save button.
 */
@FXML

```

```

private void saveButtonClicked(ActionEvent event) {
    try {
        boolean allExternallyAddedRowsValid = true;

        // Check if externally added rows exist
        if (DatabaseManager.isDatabaseEmpty("employeeshiftdata")) {
            saveAllDataToDatabase();
        } else if (!uploadCsvFile.isExternallyAddedRowsEmpty()) {
            for (CsvRow row : uploadCsvFile.getExternallyAddedRows()) {
                // Check validity and duplicates for externally added
rows
                if (!isValidRow(row) || isDuplicateRow(row)) {
                    allExternallyAddedRowsValid = false;
                    showAlert("אנא, בנתונים כפילות שיעש או שגויים נתחווים");
                }
            }
        }

        if (allExternallyAddedRowsValid) {
            // Clear externally added rows and edited rows map
            uploadCsvFile.clearExternallyAddedRows();
            editedRowsMap.clear();
        } else {
            // If there are invalid rows, do not proceed with
saving
            return;
        }
    }

    // Save all data to CSV file
    uploadCsvFile.writeDataToCsv(tableViewCSVData.getItems());

    // Save all data to Database
    DatabaseManager.saveCSVDataToDatabase("employeeshiftdata",
convertToStringCsv(tableViewCSVData.getItems()));

    showAlert("בהתלהה החיצוני ובקובץ הנתונים במסד נשמרו הנתונים");
}

} catch (Exception e) {
    e.printStackTrace();
    showAlert("שגיאה נסה, הנתונים שמיירת בהצלחה נזקודה");
}
}

/**
 * Checks if a given CsvRow is a duplicate in the database.
 *
 * @param newRow The CsvRow to check for duplication.
 * @return True if the row is a duplicate, false otherwise.
 */
private boolean isDuplicateRow(CsvRow newRow) {
    // Check if the combination of employee ID and date already exists
in the database
    return DatabaseManager.isDataExists(sessionFactory,
"employeeshiftdata", newRow.getEmployeeId(), newRow.getDate());
}

/**
 * Retrieves a formatted message indicating the invalid fields in a
CsvRow.

```

```

/*
 * @param row The CsvRow to check for invalid fields.
 * @return A formatted message indicating the invalid fields.
 */
private String getInvalidMessage(CsvRow row) {
    // Check if all the required fields are not empty
    StringBuilder invalidMessage = new StringBuilder("אף אחד לא נמצא\nבבאים הבאים:\n");
    invalidMessage.append("השעות סהכ\n");

    if (row.getTotalWorkHours().isEmpty()) {
        invalidMessage.append("- שעת כניסה\n");
    }
    if (row.getExitHour().isEmpty()) {
        invalidMessage.append("- שעת יציאה\n");
    }
    if (row.getStartHour().isEmpty()) {
        invalidMessage.append("- שעת סיום\n");
    }
    if (row.getDate().isEmpty()) {
        invalidMessage.append("- תאריך\n");
    }
    if (row.getEmployeeId().isEmpty()) {
        invalidMessage.append("- ID\n");
    }

    return invalidMessage.toString();
}

/**
 * Checks if a CsvRow is valid by ensuring all required fields are not
empty and the employee ID exists.
 *
 * @param row The CsvRow to check for validity.
 * @return True if the row is valid, false otherwise.
 */
private boolean isValidRow(CsvRow row) {
    // Check if all the required fields are not empty
    boolean isValid = !row.getTotalWorkHours().isEmpty() &&
        !row.getExitHour().isEmpty() &&
        !row.getStartHour().isEmpty() &&
        !row.getEmployeeId().isEmpty();

    if (!isValid) {
        showAlert(getInvalidMessage(row));
        return false;
    }

    // Check if the employee ID exists in the employee table
    if (!employeeExists(row.getEmployeeId())) {
        showAlert("לא ניתן למצוא ID: " + row.getEmployeeId() + "\n"
        + "במערכת נמצאו,"
        + "שונית ומספר שורה מפוקע.");
        return false;
    }

    return true;
}

/**
 * Checks if an employee with the given ID exists in the employee
table.
 */

```

```

        * @param employeeId The ID of the employee to check for existence.
        * @return True if the employee exists, false otherwise.
        */
    private boolean employeeExists(String employeeId) {
        // Check if the employee ID exists in the employee table
        try (Session session = sessionFactory.openSession()) {
            Query<Employee> query = session.createQuery("FROM Employee
WHERE employee_id = :employeeId", Employee.class);
            query.setParameter("employeeId", employeeId);
            return query.uniqueResult() != null;
        } catch (Exception e) {
            e.printStackTrace();
            return false;
        }
    }

    /**
     * Updates the specified CsvRow in the database with the edited values.
     *
     * @param editedRow The CsvRow with the edited values.
     */
    private void updateRowInDatabase(CsvRow editedRow) {
        try (Session session = sessionFactory.openSession()) {
            Employee currentUser =
UserSession.getInstance().getCurrentUser();

            if (currentUser == null) {
                handleCurrentUserNull();
                return;
            }

            Transaction transaction = null;

            try {
                transaction = session.beginTransaction();
                String updateQuery = "UPDATE employeeshiftdata SET " +
                    "total_work_hours = :totalWorkHours, " +
                    "break_time = :breakTime, " +
                    "end_of_shift = :exitHour, " +
                    "start_of_shift = :startHour, " +
                    "date = :dateTable, " +
                    "employee_id = :employeeId " +
                    "WHERE composite_key = :compositeKey";

                Query<?> query = session.createNativeQuery(updateQuery);
                setQueryParameters(query, editedRow);

                // Commit the transaction
                transaction.commit();
            } catch (Exception e) {
                handleTransactionException(transaction, e);
            }
        }
    }

    /**
     * Handles the case where the current user is null during an update
     * operation.
     * Prints an error message to the console.
     */

```

```

private void handleCurrentUserNull() {
    // Handle the case where currentUser is null (perhaps log an error
    or throw an exception)
    System.err.println("Error: currentUser is null in
updateRowInDatabase");
}

/**
 * Handles exceptions that occur during a database transaction,
including rolling back the transaction if needed.
 *
 * @param transaction The transaction being executed.
 * @param e           The exception that occurred.
 */
private void handleTransactionException(Transaction transaction,
Exception e) {
    if (transaction != null && transaction.isActive()) {
        transaction.rollback();
    }
    e.printStackTrace();
}

/**
 * Sets the parameters of the given database query using the values
from the CsvRow.
 *
 * @param query The database query.
 * @param row   The CsvRow containing the values.
 */
private void setQueryParameters(Query<?> query, CsvRow row) {
    query.setParameter("totalWorkHours", row.getTotalWorkHours());
    query.setParameter("breakTime", row.getBreakTime());
    query.setParameter("exitHour", row.getExitHour());
    query.setParameter("startHour", row.getStartHour());
    query.setParameter("dateTable", row.getDate());
    query.setParameter("employeeId", row.getEmployeeId());
}

/**
 * Handles the commit of an edit event on a TableView cell.
 * Validates the new value, rolls back the edit if needed, and updates
the CSV file and database.
 *
 * @param event The CellEditEvent triggered by the edit.
 * @param column The TableColumn being edited.
 */
private void handleEditCommit(TableColumn.CellEditEvent<CsvRow, String>
event, TableColumn<CsvRow, String> column) {
    CsvRow editedRow = event.getRowValue();
    String newValue = event.getNewValue();

    // Store the old value before it's changed
    String oldValue = event.getOldValue();

    // Validate date and hours formats only for specific columns
    if (column.equals(tc_date) && !isValidDateFormat(newValue)) {
        // Rollback the edit if validation fails for date
        tableViewCSVData.getItems().set(event.getTablePosition().getRow(),
editedRow);
        return;
    }
}

```

```

        }

        if ((column.equals(tc_enterHour) || column.equals(tc_exitHour)) &&
isValidHoursFormat(newValue)) {
            // Rollback the edit if validation fails for start/end hours

tableViewCSVData.getItems().set(event.getTablePosition().getRow(),
editedRow);
            return;
        }

        // Update the specific property based on the edited column
updatePropertyBasedOnColumn(editedRow, column, newValue);

        // Check for duplicate in the TableView
if (isDuplicateRowInTableView(editedRow)) {
    showAlertDialog("ערוך אנו, זהה בתאריך בມערך רשות כבר היה הувוד"
    +" והמשך זהה מידע/תאריך לעדכן כדי הקודם התאריך את מזק או אחר לתאריך"
    +". בפערות");
}

        // Rollback the edit to prevent adding a duplicate row

tableViewCSVData.getItems().set(event.getTablePosition().getRow(),
editedRow);
        return;
    }

    // Save changes to the CSV file and the database
try {
    uploadCsvFile.updateRowInCsv(editedRow);
    updateRowInDatabase(editedRow);
} catch (IOException e) {
    throw new RuntimeException(e);
}
}

/**
 * Checks if a CsvRow is a duplicate in the TableView.
 *
 * @param editedRow The CsvRow being checked for duplication.
 * @return True if the row is a duplicate, false otherwise.
 */
private boolean isDuplicateRowInTableView(CsvRow editedRow) {
    // Check if the combination of date and employee ID already exists
in the TableView
    return tableViewCSVData.getItems().stream()
        .anyMatch(row -> row != editedRow &&
row.getDate().equals(editedRow.getDate())
        &&
row.getEmployeeId().equals(editedRow.getEmployeeId()));
}

/**
 * Checks if the provided date string is in a valid date format
(dd/mm/yyyy).
 *
 * @param date The date string to validate.
 * @return True if the date is in a valid format, false otherwise.
 */
private boolean isValidDateFormat(String date) {
    // Define the expected date format
}

```

```

String dateFormatPattern = "\\d{2}/\\d{2}/\\d{4}";

// Check if the date matches the expected format
if (!date.matches(dateFormatPattern)) {
    showAlert("dd/mm/yyyy אונא, שגוי תאריך פורטט :");
    return false;
}

return true;
}

/**
 * Checks if the provided hours string is in a valid hours format
(h:mm).
 *
 * @param hours The hours string to validate.
 * @return True if the hours are in a valid format, false otherwise.
 */
private boolean isValidHoursFormat(String hours) {
    // Define the expected hours format
    String hoursFormatPattern = "\\d{1,2}:\\d{2}";

    // Check if the hours match the expected format
    if (!hours.matches(hoursFormatPattern)) {
        showAlert("הבא בפורטט הכנס אונא שגוי שעת פורטט: h:mm");
        return false;
    }

    return true;
}

/**
 * Updates the specific property of a CsvRow based on the edited
TableColumn and new value.
 *
 * @param editedRow      The CsvRow being updated.
 * @param editedColumn   The TableColumn being edited.
 * @param newValue       The new value for the property.
 */
private void updatePropertyBasedOnColumn(CsvRow editedRow,
TableColumn<CsvRow, String> editedColumn, String newValue) {
    // Update the specific property based on the edited column
    if (editedColumn.equals(tc_employeeId)) {
        editedRow.setEmployeeId(newValue);
    } else if (editedColumn.equals(tc_date)) {
        editedRow.setDate(newValue);
    } else if (editedColumn.equals(tc_exitHour)) {
        editedRow.setExitHour(newValue);
    } else if (editedColumn.equals(tc_enterHour)) {
        editedRow.setStartHour(newValue);
    } else if (editedColumn.equals(tc_breakTime)) {
        editedRow.setBreakTime(newValue);
    } else if (editedColumn.equals(tc_totalWorkHrs)) {
        editedRow.setTotalWorkHours(newValue);
    }
}

/**
 * Saves all data from the TableView to the database.
 */
private void saveAllDataToDatabase() {

```

```

        ObservableList<CsvRow> csvRows = tableViewCSVData.getItems();
        DatabaseManager.saveCSVDataToDatabase("employeeshiftdata",
convertToStringCsv(csvRows));
    }

    /**
     * Handles the button click event to navigate to the main employee
management page.
     *
     * @param event The ActionEvent triggered by the button click.
     * @throws IOException If an I/O exception occurs during navigation.
     */
@FXML
void mainPageButtonClicked(ActionEvent event) throws IOException {
    navigateToManageEmployeePage(event);
}

    /**
     * Handles the button click event to perform a search based on the
entered employee ID.
     *
     * @param event The ActionEvent triggered by the button click.
     */
@FXML
private void searchButtonClicked(ActionEvent event) {
    String employeeId = tf_employeeID.getText().trim();

    if (!employeeId.isEmpty() && employeeId.matches("^[\\d+]$")) {
        loadLastSavedDataByEmployeeIdIntoTableview(employeeId);
    } else {
        showAlert("מספר זהב חייב להיות 9 ספרות");
    }
}

    /**
     * Loads the last saved data from the database based on the provided
employee ID into the TableView.
     *
     * @param employeeId The employee ID for which to load the data.
     */
private void loadLastSavedDataByEmployeeIdIntoTableview(String
employeeId) {
    List<CsvRow> data =
DatabaseManager.loadLastSavedDataByEmployeeId("employeeshiftdata",
CsvRow.class, employeeId);

    if (data.isEmpty()) {
        showAlert("לא ניתן למצוא נתונים");
    } else {
        ObservableList<CsvRow> dataModels =
FXCollections.observableArrayList(data);
        tableViewCSVData.setItems(dataModels);
    }
}

```

```

package com.example.hrpulse.Controllers.EmployeeController;

import com.example.hrpulse.Services.DatabaseManager;
import com.example.hrpulse.Services.Interfaces.EmployeeNavigators;
import com.example.hrpulse.Services.Objects.Employee;
import com.example.hrpulse.Session.UserSession;
import javafx.collections.ObservableList;
import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.scene.control.*;
import org.hibernate.SessionFactory;

import javax.mail.*;
import javax.mail.internet.InternetAddress;
import javax.mail.internet.MimeMessage;
import java.io.IOException;
import java.util.List;
import java.util.Properties;
import java.util.stream.Collectors;

import static
com.example.hrpulse.Services.DatabaseManager.retrieveEmployees;

/**
 * The `feedBackEmployeeController` class manages the user interface for
providing feedback to employees.
 */
public class feedBackEmployeeController implements EmployeeNavigators {

    // Constructors to allow for different ways of initializing the
controller
    private DatabaseManager databaseManager;
    private SessionFactory sessionFactory;

    @FXML
    private Button backButtonClicked;

    @FXML
    private ListView<String> lst_employee;

    @FXML
    private TextField label_message;

    @FXML
    private Button send_button;

    @FXML
    public void initialize() {
        // Initialization logic for the controller
        List<Employee> employeeList = retrieveEmployees();
        List<String> employeeNames = employeeList.stream()
            .map(employee -> employee.getFirstName() + " " +
employee.getLastName() + " :" + employee.getEmployeeId())
            .collect(Collectors.toList());

        lst_employee.getItems().addAll(employeeNames);
    }

    @FXML
    void backButton(ActionEvent event) throws IOException {

```

```

        // Method to handle the back button click
        navigateToManageEmployeePage(event);
    }

    @FXML
    void sendButtonClicked(ActionEvent event) {
        // Method to handle sending feedback
        // get the current user from UserSession
        UserSession userSession = UserSession.getInstance();
        Employee currentUser = userSession.getCurrentUser();
        // Get the selected employee from the list view
        MultipleSelectionModel<String> selectionModel =
lst_employee.getSelectionModel();
        ObservableList<String> selectedEmployees =
selectionModel.getSelectedItems();

        if (selectedEmployees.isEmpty()) {
            // No employee selected, handle this case accordingly
            showConfirmationDialog("הנחייה לאישר צוות");
            System.out.println("צוות לא נבחר");
            return;
        }
        // Assuming your Employee class has an getEmail() method
        String selectedEmployeeId =
selectedEmployees.get(0).split(":")[1].trim();
        String selectedEmployeeEmail = getEmailById(selectedEmployeeId,
retrieveEmployees());

        String userInput = label_message.getText();
        String messageContent = "this is a message from " +
currentUser.getFirstName() + " " + currentUser.getLastName() + "\n" +
userInput;

        // Send the email using JavaMail
        sendEmail(selectedEmployeeEmail, "Feedback from HR Pulse",
messageContent);

        System.out.println(messageContent);
        showConfirmationDialog("הצגתה נשלחה");
        System.out.println("הצגתה נשלחה");
    }

    private void sendEmail(String selectedEmployeeEmail, String
feedback_from_hr_pulse, String messageContent) {
    // Method to send an email using JavaMail
    String host = "smtp.gmail.com";
    String username = "massriabdallahdev@gmail.com";
    String password = "fqtc dvkj qqqs tsip";

    Properties properties = new Properties();
    properties.put("mail.smtp.auth", "true");
    properties.put("mail.smtp.starttls.enable", "true");
    properties.put("mail.smtp.host", host);
    properties.put("mail.smtp.port", "587");

    Session session = Session.getDefaultInstance(properties, new
javax.mail.Authenticator() {
        protected PasswordAuthentication getPasswordAuthentication() {
            return new PasswordAuthentication(username, password);
        }
    });
}

```

```

        session.setDebug(true);
        try {
            Message message = new MimeMessage(session);
            message.setFrom(new
InternetAddress("massriabdallahdev@gmail.com"));
            message.setRecipient(Message.RecipientType.TO, new
InternetAddress(selectedEmployeeEmail));
            message.setSubject(feedback_from_hr_pulse);
            message.setText(messageContent);
            Transport.send(message);
        } catch (Exception e) {
            System.out.println("הארת שגיאה : " + e.getMessage());
        }
    }

    private String getEmailById(String selectedEmployeeId, List<Employee>
retrieveEmployees) {
    // Method to get an employee's email by their ID
    String email = "";
    for (Employee value : retrieveEmployees
    ) {
        if (value.getEmployeeId() ==
Integer.parseInt(selectedEmployeeId))
            email = value.getEmail();
    }
    return email;
}

private void showConfirmationDialog(String message) {
    // Helper method to display a confirmation dialog
    Alert alert = new Alert(Alert.AlertType.INFORMATION);
    alert.setTitle("ביטול או אישור");
    alert.setHeaderText(null);
    alert.setContentText(message);
    alert.showAndWait();
}
}

package com.example.hrpulse.Controllers.EmployeeController;

import com.example.hrpulse.Services.Interfaces.EmployeeNavigators;
import com.example.hrpulse.Services.Interfaces.Navigators;
import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import java.io.IOException;

/**
 * The `ManageEmployeeController` class manages the user interface for
managing employees.
 */
public class ManageEmployeeController implements Navigators,
EmployeeNavigators {

    /**
     * Handles the action when the "Add Employee" button is clicked.
     *
     * @param event The ActionEvent triggered by the user.
     * @throws IOException If an I/O error occurs.

```

```

    */
@FXML
void addEmployeeClicked(ActionEvent event) throws IOException {
    navigateToCreateEmployeePage(event);
}

/**
 * Handles the action when the "Back to Manager Page" button is clicked.
 *
 * @param event The ActionEvent triggered by the user.
 * @throws IOException If an I/O error occurs.
 */
@FXML
void backToManagerPage(ActionEvent event) throws IOException {
    navigateToMainPage(event);
}

/**
 * Handles the action when the "Edit Shift" button is clicked.
 *
 * @param event The ActionEvent triggered by the user.
 * @throws IOException If an I/O error occurs.
 */
@FXML
void editShiftClicked(ActionEvent event) throws IOException {
    navigateToEditEmployeeShiftPage(event);
}

/**
 * Handles the action when the "Feedback" button is clicked.
 *
 * @param event The ActionEvent triggered by the user.
 * @throws IOException If an I/O error occurs.
 */
@FXML
void feedbackClicked(ActionEvent event) throws IOException {
    navigateToFeedbackEmployeePage(event);
}

/**
 * Handles the action when the "Edit Employee" button is clicked.
 *
 * @param event The ActionEvent triggered by the user.
 * @throws IOException If an I/O error occurs.
 */
@FXML
void editEmployeeClicked(ActionEvent event) throws IOException {
    navigateToEditEmployeePage(event);
}
}

```

Reports controllers:

```
package com.example.hrpulse.Controllers.ReportsControllers;

import com.example.hrpulse.Services.Database.DatabaseManager;
import com.example.hrpulse.Services.Interfaces.ReportsNavigators;
import com.example.hrpulse.Services.Objects.Employee;
import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.scene.control.*;
import javafx.scene.layout.GridPane;
import net.sf.jasperreports.engine.*;
import net.sf.jasperreports.engine.design.JRDesignQuery;
import net.sf.jasperreports.engine.design.JasperDesign;
import net.sf.jasperreports.engine.xml.JRXmlLoader;
import net.sf.jasperreports.view.JasperViewer;
import org.hibernate.SessionFactory;
import javafx.collections.FXCollections;
import javafx.collections.ObservableList;

import java.io.IOException;
import java.sql.*;
import java.util.List;

import static
com.example.hrpulse.Services.Database.DatabaseManager.retrieveEmployees;

public class MonthlyShiftEmployeeController implements ReportsNavigators {
    private DatabaseManager databaseManager;
    private SessionFactory sessionFactory;

    public MonthlyShiftEmployeeController() {
    }

    public MonthlyShiftEmployeeController(DatabaseManager databaseManager) {
        this.databaseManager = databaseManager;
        this.sessionFactory = null;
    }

    public MonthlyShiftEmployeeController(DatabaseManager databaseManager,
SessionFactory sessionFactory) {
        this.databaseManager = databaseManager;
        this.sessionFactory = sessionFactory;
    }

    @FXML
    private ChoiceBox<Integer> cb_monthSelect;

    @FXML
    private ListView<Employee> lv_retrieveEmployees;
    private Integer selectedMonth;
    private Employee selectedEmployee;
    private ObservableList<Employee> employeeObservableList;

    @FXML
```

```

public void initialize() {
    // Populate the ChoiceBox with months
    ObservableList<Integer> months =
        FXCollections.observableArrayList(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12);
    cb_monthSelect.setItems(months);

    // Initialize the employee list
    List<Employee> employeeList = retrieveEmployees();
    employeeObservableList =
        FXCollections.observableArrayList(employeeList);

    // Set up the ListView to display the employee details
    lv_retrieveEmployees.setItems(employeeObservableList);
    lv_retrieveEmployees.setCellFactory(param -> new
ListCell<Employee>() {
    @Override
    protected void updateItem(Employee item, boolean empty) {
        super.updateItem(item, empty);
        if (empty || item == null) {
            setText(null);
        } else {
            GridPane gridPane = new GridPane();
            gridPane.setHgap(10);

            Label firstNameLabel = new Label("שם פרטי:");
            Label lastNameLabel = new Label("שם משפחה:");
            Label idLabel = new Label("מספר ת.ז:");

            Label firstNameValue = new Label(item.getFirstName());
            Label lastNameValue = new Label(item.getLastName());
            Label idValue = new
Label(String.valueOf(item.getEmployeeId()));

            gridPane.addColumn(0, firstNameLabel, lastNameLabel,
idLabel);
            gridPane.addColumn(1, firstNameValue, lastNameValue,
idValue);

            setText(null);
            setGraphic(gridPane);
        }
    }
});
    // Set up the layout direction to right-to-left
    lv_retrieveEmployees.setStyle("-fx-alignment: CENTER-LEFT;");

    // Set up the selection model for the ListView

    lv_retrieveEmployees.getSelectionModel().selectedItemProperty().addListener(
        (observable, oldValue, newValue) -> {
            selectedEmployee = newValue;
        });
    // Set up the selection model for the ChoiceBox

    cb_monthSelect.getSelectionModel().selectedItemProperty().addListener((obse
rvable, oldValue, newValue) -> {
        selectedMonth = newValue;
    });
}

```

```

public Integer getSelectedMonth() {
    return selectedMonth;
}

public Employee getSelectedEmployee() {
    return selectedEmployee;
}

@FXML
void back_btn(ActionEvent event) throws IOException {
    navigateToProductionOfReportsPage(event);
}

@FXML
void showListClicked(ActionEvent event) {
    if (selectedMonth == null || selectedEmployee == null) {
        showAlert("לא המשיכך בצד ימין שעובד בחודש אונאי");
        return;
    }

    try {
        // Load the MySQL JDBC driver
        Class.forName("com.mysql.jdbc.Driver");

        // Establish a connection to the MySQL database
        Connection connection =
DriverManager.getConnection("jdbc:mysql://localhost:3306/pulsedb", "root",
"hrpulse123");

        // Path to the JR XML file (JasperReports XML template)
        String reportPath = "emo2.jrxml";

        JasperDesign jd = JRXmlLoader.load(reportPath);
        Integer employeeId = selectedEmployee.getEmployeeId();

        // Modified SQL query to include the selected month
        String sql = "SELECT e.employee_id AS employee_id, " +
                    "e.first_name, " +
                    "e.last_name, " +
                    "STR_TO_DATE(esd.date, '%d/%m/%Y') AS formatted_date, " +
                    "esd.start_of_shift, " +
                    "esd.end_of_shift, " +
                    "esd.total_work_hours " +
                    "FROM employees e " +
                    "JOIN employeeshiftdata esd ON e.employee_id =
esd.employee_id " +
                    "WHERE e.employee_id = ? AND
MONTH(STR_TO_DATE(esd.date, '%d/%m/%Y')) = ?";
    }
}

// Using prepared statement to avoid SQL injection
try (PreparedStatement pstmt =
connection.prepareStatement(sql)) {
    pstmt.setInt(1, employeeId);
    pstmt.setInt(2, selectedMonth);

    // Execute the query
    ResultSet resultSet = pstmt.executeQuery();
}

```

```

JRDesignQuery newQuery = new JRDesignQuery();
newQuery.setText(sql);
jd.setQuery(newQuery);

// Compile the JRXML file into a JasperReport object
JasperReport jr = JasperCompileManager.compileReport(jd);

// Fill the report with data and create a JasperPrint
object
JasperPrint jp = JasperFillManager.fillReport(jr, null, new
JRResultSetDataSource(resultSet));

// View the JasperPrint object using JasperViewer
JasperViewer.viewReport(jp, false);

// Close the database connection
connection.close();
}
} catch (ClassNotFoundException | SQLException | JRException e) {
    throw new RuntimeException(e);
}
}

// Helper method to show an alert
private void showAlert(String message) {
    Alert alert = new Alert(Alert.AlertType.ERROR, message,
ButtonType.OK);
    alert.showAndWait();
}

}

package com.example.hrpulse.Controllers.ReportsControllers;

import com.example.hrpulse.Services.Interfaces.Navigators;
import com.example.hrpulse.Services.Interfaces.ReportsNavigators;
import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.scene.control.Button;
import net.sf.jasperreports.engine.*;
import net.sf.jasperreports.view.JasperViewer;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import java.io.IOException;

/**
 * The `productionOfReportsController` class manages the user interface for
generating reports.
 */
public class productionOfReportsController implements Navigators,
ReportsNavigators {

@FXML
private Button back_btn;

```

```

@FXML
private Button employee_reports_btn;

@FXML
private Button shiftEmployee_reports_btn;

@FXML
void backToMainClicked(ActionEvent event) throws IOException {
    // Method to handle the "Back to Main" button click
    navigateToMainPage(event);
}

@FXML
void shiftEmployeeReportsClicked(ActionEvent event) throws IOException
{
    navigateToMonthlyShiftEmployee(event);
}

@FXML
void employeeRepotsClicked(ActionEvent event) {
    try {
        // Load the MySQL JDBC driver
        Class.forName("com.mysql.jdbc.Driver");

        // Establish a connection to the MySQL database
        Connection connection =
DriverManager.getConnection("jdbc:mysql://localhost:3306/pulsedb", "root",
"hrpulse123");

        // Path to the JR XML file (JasperReports XML template)
        String reportPath = "EmployeeReport.jrxml";

        // Compile the JRXML file into a JasperReport object
        JasperReport jr =
JasperCompileManager.compileReport(reportPath);

        // Fill the report with data and create a JasperPrint object
        JasperPrint jp = JasperFillManager.fillReport(jr, null,
connection);

        // View the JasperPrint object using JasperViewer
        JasperViewer.viewReport(jp, false);

        // Close the database connection
        connection.close();

    } catch (ClassNotFoundException | SQLException | JRException e) {
        throw new RuntimeException(e);
    }
}
}

```

UserController and login controller:

```
package com.example.hrpulse.Controllers.UsersControllers;

import com.example.hrpulse.Services.Interfaces.EmployeeNavigators;
import com.example.hrpulse.Services.Interfaces.Navigators;
import com.example.hrpulse.Services.Interfaces.ReportsNavigators;
import com.example.hrpulse.Services.Objects.Employee;
import com.example.hrpulse.Session.UserSession;
import javafx.application.Platform;
import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.scene.control.Button;
import javafx.scene.control.Label;
import javafx.scene.layout.BorderPane;

import java.io.IOException;

/**
 * The `UserController` class manages the user interface for the logged-in
 * user.
 */
public class UserController implements Navigators, EmployeeNavigators,
ReportsNavigators {

    @FXML
    private Label label_userName;

    @FXML
    private Button reportsButton;

    @FXML
    private Button EditEmployeeButton;

    @FXML
    private BorderPane borderPane;

    /**
     * Initializes the controller. Sets up the UI based on the logged-in
     * user's role.
     */
    public void initialize() {
        // Get the current user from the UserSession
        UserSession userSession = UserSession.getInstance();
        Employee currentUser = userSession.getCurrentUser();

        if (currentUser != null) {
            // Set the text of the label_userName to the user's name
            label_userName.setText("User: " + currentUser.getFirstName());

            // Apply different styles based on the user's role
            if ("headOfDepartment".equals(currentUser.getEmployeeRole())) {
                // Head of Department role
                applyHeadOfDepartmentStyles();
            } else if ("secretary".equals(currentUser.getEmployeeRole())) {
                // Secretary role
                applySecretaryStyles();
            } else if ("manager".equals(currentUser.getEmployeeRole())) {
                // Manager role
            }
        }
    }
}
```

```

        applyManagerStyles();
    }
}

private void applyHeadOfDepartmentStyles() {
    // Remove existing style classes to prevent conflicts
    borderPane.getStyleClass().removeAll("secretary");
    // Add the headOfDepartment style class & Hide fields
    borderPane.getStyleClass().add("headOfDepartment");
    hideHeadOfDepartmentButtons();
}

private void hideHeadOfDepartmentButtons() {
    EditEmployeeButton.setVisible(false);
}

private void applySecretaryStyles() {
    // Remove existing style classes to prevent conflicts
    borderPane.getStyleClass().removeAll("headOfDepartment");
    // Add the secretary style class & Hide fields
    borderPane.getStyleClass().add("secretary");
    hideSecretaryButtons();
}

private void hideSecretaryButtons() {
    EditEmployeeButton.setVisible(false);
    reportsButton.setVisible(false);
}

private void applyManagerStyles() {
    // Remove existing style classes to prevent conflicts
    borderPane.getStyleClass().removeAll("headOfDepartment");
    // Add the manager style class
    borderPane.getStyleClass().add("manager");
}

@FXML
void reportsClicked(ActionEvent event) throws IOException {
    // Method to handle the "Reports" button click
    navigateToProductionOfReportsPage(event);
}

@FXML
void manageEmployeeClicked(ActionEvent event) throws IOException {
    // Method to handle the "Manage Employee" button click
    navigateToManageEmployeePage(event);
}

@FXML
void NavigateToManageDepartment(ActionEvent event) throws IOException {
    // Method to handle the "Manage Department" button click
    navigateToManageDepartment(event);
}

@FXML
void NavigateToReportsPage(ActionEvent event) throws IOException {
}

```

```

        // Show error or display a message for the user; this button will
be used in the next version
    }

    @FXML
    void ExitButtonClicked(ActionEvent event) {
        // Method to handle the "Exit" button click
        Platform.exit();
    }

    @FXML
    void HomePageClicked(ActionEvent event) throws IOException {
        // Method to handle the "Home Page" button click
        navigateToLoginPage(event);
    }
}

```

```

package com.example.hrpulse.Controllers;

import com.example.hrpulse.Services.Database.DatabaseManager;
import com.example.hrpulse.Services.Objects.Employee;
import com.example.hrpulse.Services.Interfaces.Navigators;
import com.example.hrpulse.Session.UserSession;
import javafx.application.Platform;
import javafx.concurrent.Task;
import javafx.concurrent.WorkerStateEvent;
import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.scene.control.Label;
import javafx.scene.control.PasswordField;
import javafx.scene.control.TextField;
import javafx.util.Duration;
import org.hibernate.SessionFactory;

import java.io.IOException;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import java.util.concurrent.Executors;
import java.util.concurrent.ScheduledExecutorService;
import java.util.concurrent.TimeUnit;

import static
com.example.hrpulse.Services.Database.DatabaseManager.retrieveEmployees;

/**
 * The `loginController` class manages the user interface for the login
functionality.
 */
public class loginController implements Navigators {
    private DatabaseManager databaseManager;
    private SessionFactory sessionFactory;

```

```

@FXML
private Label wrongLogin;

private int loginAttempts = 0;
private final int maxLoginAttempts = 3;
private final Duration lockoutDuration = Duration.minutes(5);
private ScheduledExecutorService scheduler =
Executors.newScheduledThreadPool(1);

public loginController() {
    // Default constructor
}

public loginController(DatabaseManager databaseManager) {
    this.databaseManager = databaseManager;
    this.sessionFactory = null;
}

public loginController(DatabaseManager databaseManager, SessionFactory sessionFactory) {
    this.databaseManager = databaseManager;
    this.sessionFactory = sessionFactory;
}

// Hard coded list of employees - to be replaced with a database /
bypass the database login
private static final Map<String, Employee> employees = new HashMap<>();

static {
    employees.put("admin_m", new Employee("manager",
"Globalvpsm@gmail.com", "0535216838", "1234", "manager"));
    employees.put("admin_s", new Employee("secretary",
"Globalvpsm@gmail.com", "0535216838", "1234", "secretary"));
    employees.put("admin_h", new Employee("head", "A.v.e@live.com",
"0535216838", "1234", "headOfDepartment"));
}

@FXML
private PasswordField tf_Password;

@FXML
private TextField tf_UserName;

private static final List<Employee> employeesList = getEmployees();

private static List<Employee> getEmployees() {
    return retrieveEmployees();
}

@FXML
void ExitButtonClicked() {
    Platform.exit();
}

@FXML
void userLogin(ActionEvent event) throws IOException {
    String username = tf_UserName.getText().trim().toLowerCase();
    String password = tf_Password.getText().toLowerCase();
}

```

```

        if (loginAttempts >= maxLoginAttempts) {
            wrongLogin.setText("שנית נסה אנה - כושלים כניסה מידית יותר");
            return;
        }

        // get the local employee that we declare if not found it returns
        null.
        Employee employee = employees.get(username);
        System.out.print("employee");
        System.out.println(employee);
        // get the employee from the database if not found return null.
        Employee employeeDb = getEmployeeByUsernameAndPassword(username,
password);
        System.out.print("employeeDb");
        System.out.println(employeeDb);

        if (employeeDb != null) {
            // Employee found in the database
            // Declare a new User
            UserSession.getInstance().setCurrentUser(employeeDb);
            navigateToMainPage(event);
        } else if (employee != null &&
password.equals(employee.getPassword())) {
            // Successful login
            UserSession.getInstance().setCurrentUser(employee);
            navigateToMainPage(event);
        } else {
            // Incorrect username or password
            loginAttempts++;
            if (loginAttempts >= maxLoginAttempts) {
                scheduleUnlock();
            }
            wrongLogin.setText("הש מנסה וריאם סיסמה או שגויו.");
        }
    }

private void scheduleUnlock() {
    Task<Void> unlockTask = new Task<>() {
        @Override
        protected Void call() throws Exception {
            TimeUnit.MILLISECONDS.sleep((long) lockoutDuration.toMinutes());
            loginAttempts = 0;
            return null;
        }
    };
}

unlockTask.addEventHandler(WorkerStateEvent.WORKER_STATE_SUCCEEDED,
event -> {
    wrongLogin.setText("");
});

scheduler.schedule(unlockTask, 0, TimeUnit.SECONDS);
}

private Employee getEmployeeByUsernameAndPassword(String username,
String password) {
    for (Employee employee : employeesList) {

```

```

        if (String.valueOf(employee.getEmployeeId()).equals(username)
&& employee.getPassword().equals(password) ) {
            return employee;
        }
    }
    return null;
}
}

```

CSV SERVICES:

```

package com.example.hrpulse.Services.CSV;

import com.example.hrpulse.Services.Interfaces.DataModel;
import javax.persistence.*;
import java.util.Arrays;

/**
 * Represents a row of the employee shift information.
 * Implements the DataModel interface.
 */

@Entity
@Table(name = "employeeshiftdata")
public class CsvRow implements DataModel {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    @Column(name = "compositeKey")
    private String compositeKey;

    @Column(name = "total_work_hours")
    private String totalWorkHours;

    @Column(name = "break_time")
    private String breakTime;

    @Column(name = "end_of_shift")
    private String exitHour;

    @Column(name = "start_of_shift")
    private String startHour;

    @Column(name = "date")
    private String date;

    @Column(name = "employee_id", unique = true) // Ensure uniqueness
    private String employeeId;

    /**
     * Default constructor.
     */
}

```

```

public CsvRow() {
    // Default constructor
}

/**
 * Parameterized constructor for initializing CsvRow with specified
values.
 * Sets the compositeKey based on employeeId and date.
 *
 * @param totalWorkHours The total work hours.
 * @param breakTime      The break time.
 * @param exitHour       The end of shift hour.
 * @param startHour     The start of shift hour.
 * @param date           The date of the shift.
 * @param employeeId    The employee ID.
 */
public CsvRow(String totalWorkHours, String breakTime, String exitHour,
String startHour,
            String date, String employeeId) {
    initializeFields(totalWorkHours, breakTime, exitHour, startHour,
date, employeeId);
    this.compositeKey = employeeId + "_" + date;
}

/**
 * Constructor for creating CsvRow from a String array.
 * Initialize fields with values from the array.
 * Prints an error message for invalid CSV row data.
 *
 * @param rowData The String array containing CSV row data.
 */
public CsvRow(String[] rowData) {
    if (rowData.length >= 5) {
        initializeFields(rowData[0], rowData[1], rowData[2],
rowData[3], rowData[4], "");
    } else {
        System.err.println("Invalid CSV row: " +
Arrays.toString(rowData));
    }
}

/**
 * Constructor for creating CsvRow with specified values and an empty
employeeId.
 * Initializes fields and sets the compositeKey based on employeeId and
date.
 *
 * @param totalWorkHours The total work hours.
 * @param breakTime      The break time.
 * @param exitHour       The end of shift hour.
 * @param startHour     The start of shift hour.
 * @param date           The date of the shift.
 */
public CsvRow(String totalWorkHours, String breakTime, String exitHour,
String startHour,
              String date) {
    this();
    initializeFields(totalWorkHours, breakTime, exitHour, startHour,
date, "");
}

```

```

    /**
     * Initializes the CsvRow fields with the provided values and sets
     initial values for tracking changes.
     *
     * @param totalWorkHours The total work hours.
     * @param breakTime      The break time.
     * @param exitHour       The end of shift hour.
     * @param startHour      The start of shift hour.
     * @param dateTable      The date of the shift.
     * @param employeeId     The employee ID.
     */
    private void initializeFields(String totalWorkHours, String breakTime,
String exitHour, String startHour,
                               String dateTable, String employeeId) {
        setTotalWorkHours(totalWorkHours);
        setBreakTime(breakTime);
        setExitHour(exitHour);
        setStartHour(startHour);
        setDate(dateTable);
        setEmployeeId(employeeId);
    }

    /**
     * {@inheritDoc}
     */
    @Override
    public String[] getDataAsArray() {
        return new String[]{totalWorkHours, breakTime, exitHour, startHour,
date, employeeId};
    }

    /**
     * {@inheritDoc}
     */
    @Override
    public void initializeFromCsvData(String[] data) {
        setTotalWorkHours(data[0]);
        setBreakTime(data[1]);
        setExitHour(data[2]);
        setStartHour(data[3]);
        setDate(data[4]);
        setEmployeeId(data[5]);
    }

    /**
     * Converts the CsvRow to a String array.
     */
    public String[] toArray() {
        return new String[]{
            totalWorkHours,
            breakTime,
            exitHour,
            startHour,
            date,
            employeeId,
        };
    }
}

public String getCompositeKey() {

```

```
        return employeeId + "_" + date;
    }

    public void setCompositeKey(String compositeKey) {
        this.compositeKey = compositeKey;
    }

    public String getTotalWorkHours() {
        return totalWorkHours;
    }

    public void setTotalWorkHours(String totalWorkHours) {
        this.totalWorkHours = totalWorkHours;
    }

    public String getBreakTime() {
        return breakTime;
    }

    public void setBreakTime(String breakTime) {
        this.breakTime = breakTime;
    }

    public String getExitHour() {
        return exitHour;
    }

    public void setExitHour(String exitHour) {
        this.exitHour = exitHour;
    }

    public String getStartHour() {
        return startHour;
    }

    public void setStartHour(String startHour) {
        this.startHour = startHour;
    }

    public String getDate() {
        return date;
    }

    public void setDate(String date) {
        this.date = date;
    }

    public String getEmployeeId() {
        return employeeId;
    }

    public void setEmployeeId(String employeeId) {
        this.employeeId = employeeId;
    }

    public Long getId() {
        return id;
    }

    public void setId(Long id) {
        this.id = id;
    }
```

```
    }

}

package com.example.hrpulse.Services.CSV;

import com.opencsv.CSVWriter;
import java.io.BufferedReader;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import java.util.ArrayList;
import java.util.List;

/**
 * Utility class for reading and writing CSV files using OpenCSV library.
 */
public class CsvService {

    /**
     * Reads CSV data from the specified file path.
     *
     * @param csvFilePath The path of the CSV file to read.
     * @return A list of String arrays representing the CSV data.
     * @throws IOException If an I/O error occurs.
     */
    public static List<String[]> readCsv(String csvFilePath) throws IOException {
        List<String[]> csvData = new ArrayList<>();
        if (csvFilePath.isEmpty()) {
            // Handle the case where the file path is empty.
            return csvData;
        }

        try (BufferedReader br = new BufferedReader(new FileReader(csvFilePath))) {
            String line;
            while ((line = br.readLine()) != null) {
                String[] values = line.split(",");
                for (int i = 0; i < values.length; i++) {
                    // Remove leading and trailing whitespaces and double quotes from each value
                    values[i] = values[i].trim().replaceAll("^\"|\"$", "");
                }
                csvData.add(values);
            }
        }

        return csvData;
    }

    /**
     * Writes CSV data to the specified file.
     *
     * @param filePath The path of the CSV file to write.
     * @param csvData The list of String arrays representing the CSV data.
     */
}
```

```

        */
    public static void writeCsv(String filePath, List<String[]> csvData) {
        try (CSVWriter writer = new CSVWriter(new FileWriter(filePath))) {
            writer.writeAll(csvData);
        } catch (IOException e) {
            throw new RuntimeException(e);
        }
    }
}

package com.example.hrpulse.Services.CSV;

import javafx.application.Application;
import javafx.beans.property.SimpleStringProperty;
import javafx.collections.FXCollections;
import javafx.collections.ObservableList;
import javafx.scene.Scene;
import javafx.scene.control.Alert;
import javafx.scene.control.Button;
import javafx.scene.control.TableColumn;
import javafx.scene.control.TableView;
import javafx.scene.control.cell.TextFieldTableCell;
import javafx.scene.layout.VBox;
import javafx.stage.FileChooser;
import javafx.stage.Stage;

import java.io.File;
import java.io.IOException;
import java.util.List;

/**
 * JavaFX's application for displaying and editing CSV data in a TableView.
 */
public class CSVTableView extends Application {
    public static void main(String[] args) {
        launch(args);
    }

    // ObservableList for TableView data
    ObservableList<CsvRow> data = FXCollections.observableArrayList();

    @Override
    public void start(Stage primaryStage) {
        primaryStage.setTitle("TableView");

        // TableView setup
        TableView<CsvRow> tableView = new TableView<>();
        tableView.setEditable(true); // Enable editing

        TableColumn<CsvRow, String> totalWorkHoursColumn = new
        TableColumn<>("Total Work Hours");
        totalWorkHoursColumn.setCellValueFactory(cellData -> new
        SimpleStringProperty(cellData.getValue().getTotalWorkHours()));
    }
}

```

```

totalWorkHoursColumn.setCellFactory(TextFieldTableCell.forTableColumn());
totalWorkHoursColumn.setOnEditCommit(event -> {
    event.getRowValue().setTotalWorkHours(event.getNewValue());
});

TableColumn<CsvRow, String> breakTimeColumn = new
TableColumn<>("Break time");
breakTimeColumn.setCellValueFactory(cellData -> new
SimpleStringProperty(cellData.getValue().getBreakTime()));

breakTimeColumn.setCellFactory(TextFieldTableCell.forTableColumn());
breakTimeColumn.setOnEditCommit(event -> {
    event.getRowValue().setBreakTime(event.getNewValue());
});

TableColumn<CsvRow, String> endTimeColumn = new TableColumn<>("End
time");
endTimeColumn.setCellValueFactory(cellData -> new
SimpleStringProperty(cellData.getValue().getExitHour()));
endTimeColumn.setCellFactory(TextFieldTableCell.forTableColumn());
endTimeColumn.setOnEditCommit(event -> {
    event.getRowValue().setExitHour(event.getNewValue());
});

TableColumn<CsvRow, String> startTimeColumn = new
TableColumn<>("Start time");
startTimeColumn.setCellValueFactory(cellData -> new
SimpleStringProperty(cellData.getValue().getStartHour()));

startTimeColumn.setCellFactory(TextFieldTableCell.forTableColumn());
startTimeColumn.setOnEditCommit(event -> {
    event.getRowValue().setStartHour(event.getNewValue());
});

TableColumn<CsvRow, String> dateColumn = new TableColumn<>("Date");
dateColumn.setCellValueFactory(cellData -> new
SimpleStringProperty(cellData.getValue().getDate()));
dateColumn.setCellFactory(TextFieldTableCell.forTableColumn());
dateColumn.setOnEditCommit(event -> {
    event.getRowValue(). setDate(event.getNewValue());
});

TableColumn<CsvRow, String> empIdColumn = new
TableColumn<>("EmployeeID");
empIdColumn.setCellValueFactory(cellData -> new
SimpleStringProperty(cellData.getValue().getEmployeeId()));
empIdColumn.setCellFactory(TextFieldTableCell.forTableColumn());
empIdColumn.setOnEditCommit(event -> {
    event.getRowValue().setEmployeeId(event.getNewValue());
});

// Adding columns to the TableView
tableView.getColumns().addAll(totalWorkHoursColumn,
breakTimeColumn, endTimeColumn, startTimeColumn, dateColumn, empIdColumn);
// Set data to the TableView
tableView.setItems(data);

// Buttons for actions
Button loadButton = new Button("Load CSV");
Button addButton = new Button("Add Row");

```

```

        // Set actions for buttons
        addButton.setOnAction(e -> addNewRow(tableView));

        loadButton.setOnAction(e -> {
            // FileChooser setup
            FileChooser fileChooser = new FileChooser();
            fileChooser.getExtensionFilters().add(new
FileChooser.ExtensionFilter("CSV Files", "*.csv"));
            File selectedFile = fileChooser.showOpenDialog(primaryStage);

            if (selectedFile != null) {
                try {
                    // Read CSV data and update the TableView
                    List<String[]> csvData =
CsvService.readCsv(selectedFile.getAbsolutePath());
                    data.clear(); // Clear existing data
                    for (String[] row : csvData) {
                        CsvRow newRow = new CsvRow(row[0], row[1], row[2],
row[3], row[4], row[5]);
                        data.add(newRow);
                    }
                } catch (IOException ex) {
                    showErrorDialog("Error loading CSV", "An error occurred
while loading the CSV file.");
                }
            }
        });

        // Layout setup
        VBox vbox = new VBox(tableView, loadButton);
        Scene scene = new Scene(vbox, 600, 400);
        primaryStage.setScene(scene);
        primaryStage.show();
    }

    /**
     * Display an error dialog with the specified title and content.
     *
     * @param title The title of the error dialog.
     * @param content The content of the error dialog.
     */
    private void showErrorDialog(String title, String content) {
        Alert alert = new Alert(Alert.AlertType.ERROR);
        alert.setTitle(title);
        alert.setHeaderText(null);
        alert.setContentText(content);
        alert.showAndWait();
    }

    /**
     * Add a new row to the TableView.
     *
     * @param tableView The TableView to which the row should be added.
     */
    private void addNewRow(TableView<CsvRow> tableView) {
        // Create a new row with default values or empty values
        CsvRow newRow = new CsvRow("", "", "", "", "", "");
        data.add(newRow);
        tableView.scrollTo(newRow); // Scroll to the new row
        tableView.getSelectionModel().select(newRow); // Select the new row
    }
}

```

```

        for editing
            tableView.edit(tableView.getItems().indexOf(newRow),
tableView.getColumns().get(0)); // Start editing the first cell
        }

    }

package com.example.hrpulse.Services.CSV;

import com.example.hrpulse.Services.Database.DatabaseManager;
import com.opencsv.CSVWriter;
import javafx.collections.FXCollections;
import javafx.collections.ObservableList;
import javafx.scene.control.TableView;
import javafx.stage.FileChooser;
import javafx.stage.Stage;

import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.List;

import static com.example.hrpulse.HR_Pulse.sessionFactory;
import static com.example.hrpulse.Services.CSV.CsvService.readCsv;
import static com.example.hrpulse.Services.CSV.CsvService.writeCsv;

/**
 * Utility class for uploading CSV files and managing data in a TableView.
 */
public class UploadCsvFile {

    // Fields for managing TableView, file paths, and additional rows
    private TableView<CsvRow> tableView;
    private String filePath;
    private String pulseDB;
    private boolean isTableViewLoaded = false;

    // Default CSV file path
    private String csvFilePath = "C:\\CSV file for project HR-Pulse\\employees - in-out.csv";

    private ObservableList<CsvRow> additionalRows =
    FXCollections.observableArrayList();
    private List<CsvRow> externallyAddedRows = new ArrayList<>();

    // Constructor
    public UploadCsvFile(TableView<CsvRow> tableView, String pulseDB) {
        this.tableView = tableView;
        this.pulseDB = pulseDB;
    }

    // Getter for file path
    public String getFilePath() {
        return filePath;
    }
}

```

```

    /**
     * Uploads data from a CSV file to the TableView.
     */
    public void upload() {
        // FileChooser for selecting CSV file
        FileChooser fileChooser = new FileChooser();
        fileChooser.getExtensionFilters().add(new
FileChooser.ExtensionFilter("CSV Files", "*.csv"));
        File selectedFile = fileChooser.showOpenDialog(new Stage());

        // If a file is selected
        if (selectedFile != null) {
            filePath = selectedFile.getAbsolutePath();
            csvFilePath = filePath; // Set csvFilePath to the selected file
path

            try {
                // Read CSV data and convert to CsvRows
                List<String[]> csvData = readCsv(filePath);
                ObservableList<CsvRow> csvRows = convertToCsvRows(csvData);

                // Identify externally added rows
                for (String[] rowData : csvData) {
                    String employeeId = rowData[5];
                    String date = rowData[4];

                    if (!DatabaseManager.isDataExists(sessionFactory,
pulseDB, employeeId, date)) {
                        CsvRow csvRow = new CsvRow(rowData);
                        externallyAddedRows.add(csvRow);
                    }
                }

                // Set TableView items and mark as loaded
                tableView.setItems(csvRows);
                isTableViewLoaded = true;
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
    }

    // Update a row in the CSV file and the TableView
    public void updateRowInCsv(CsvRow updatedRow) throws IOException {
        ObservableList<CsvRow> data = tableView.getItems();
        int index = data.indexOf(updatedRow);

        if (index != -1) {
            data.set(index, updatedRow);
            writeDataToCsv(data);
        } else {
            System.out.println("Row not found in the TableView. Cannot
update CSV.");
        }
    }

    // Write data to the CSV file
    public void writeDataToCsv(ObservableList<CsvRow> data) throws
IOException {
        try (CSVWriter writer = new CSVWriter(new FileWriter(csvFilePath)))
{

```

```

        for (CsvRow row : data) {
            writer.writeNext(new String[]{
                row.getTotalWorkHours(),
                row.getBreakTime(),
                row.getExitHour(),
                row.getStartHour(),
                row.getDate(),
                row.getEmployeeId(),
            });
        }
    }

    // Check if TableView is loaded
    public boolean isTableViewLoaded() {
        return isTableViewLoaded;
    }

    /**
     * Removes a row from the CSV file and the database.
     */
    public void removeRowFromCsvAndDatabase(CsvRow row) throws IOException
    {
        List<String[]> csvData = readCsv(csvFilePath);
        csvData.removeIf(csvRow -> Arrays.equals(csvRow, row.toArray()));
        writeCsv(csvFilePath, csvData);
        DatabaseManager.deleteRowFromDatabase("employeeshiftdata",
        row.getEmployeeId(), row.getDate());
    }

    // Convert a list of String arrays to CsvRows
    public ObservableList<CsvRow> convertToCsvRows(List<String[]> csvData)
    {
        ObservableList<CsvRow> csvRows =
FXCollections.observableArrayList();
        for (String[] row : csvData) {
            if (row.length >= 5) {
                CsvRow csvRow = new CsvRow(
                    row[0], row[1], row[2], row[3], row[4], row[5]);
                csvRows.add(csvRow);
            } else {
                System.err.println("Invalid CSV row: " +
Arrays.toString(row));
            }
        }
        return csvRows;
    }

    // Mark a row as externally added
    public void markExternallyAddedRow(CsvRow row) {
        externallyAddedRows.add(row);
    }

    // Retrieve externally added rows
    public List<CsvRow> getExternallyAddedRows() {
        return new ArrayList<>(externallyAddedRows);
    }

    // Check if externally added rows are empty
    public boolean isExternallyAddedRowsEmpty() {
        return externallyAddedRows.isEmpty();
    }
}

```

```

    }

    // Clear externally added rows
    public void clearExternallyAddedRows() {
        externallyAddedRows.clear();
    }
}

```

HIBERNATE.CCFG.XML

```

<!DOCTYPE hibernate-configuration PUBLIC "-//Hibernate/Hibernate
Configuration DTD 3.0//EN"
      "http://hibernate.sourceforge.net/hibernate-configuration-3.0.dtd">
<hibernate-configuration>
    <session-factory>
        <!-- Database connection settings -->
        <property
name="hibernate.connection.driver_class">com.mysql.cj.jdbc.Driver</property>
        <property
name="hibernate.connection.url">jdbc:mysql://localhost:3306/pulsedb?serverT
imezone=UTC</property>
        <property name="hibernate.connection.username">root</property>
        <property
name="hibernate.connection.password">hrpulse123</property>

        <!-- Specify dialect -->
        <property
name="hibernate.dialect">org.hibernate.dialect.MySQL8Dialect</property>

        <!-- Enable Hibernate's automatic session context management -->
        <property
name="hibernate.current_session_context_class">thread</property>

        <!-- Enable Hibernate's automatic session management -->
        <property
name="hibernate.transaction.coordinator_class">org.hibernate.transaction.JD
BCTransactionFactory</property>

        <!-- Echo all executed SQL to stdout -->
        <property name="hibernate.show_sql">true</property>

        <!-- Specify the packages to scan for annotated entities -->
        <property name="hibernate.archive.autodetect">class</property>
        <property
name="hibernate.packageToScan">com.example.hrpulse.Services.CSV</property>

        <!-- Mapping for annotated entity -->
        <mapping class="com.example.hrpulse.Services.CSV.CsvRow"/>
    </session-factory>
</hibernate-configuration>

```

Module.info

```
module com.example.hrpulse {  
    requires javafx.controls;  
    requires javafx.fxml;  
    requires opencsv;  
    requires org.hibernate.orm.core;  
    requires java.sql;  
    requires java.mail;  
    requires java.naming;  
    requires java.persistence;  
    requires jasperreports;  
    requires mysql.connector.j;  
  
    opens com.example.hrpulse to javafx.fxml;  
    exports com.example.hrpulse;  
    exports com.example.hrpulse.Controllers.UsersControllers;  
    opens com.example.hrpulse.Controllers.UsersControllers to javafx.fxml;  
    exports com.example.hrpulse.Controllers;  
    opens com.example.hrpulse.Controllers to javafx.fxml;  
    exports com.example.hrpulse.Controllers.DepartmentController;  
    opens com.example.hrpulse.Controllers.DepartmentController to  
javafx.fxml;  
    exports com.example.hrpulse.Controllers.EmployeeController;  
    opens com.example.hrpulse.Controllers.EmployeeController to  
javafx.fxml;  
    exports com.example.hrpulse.Controllers.ReportsControllers;  
    opens com.example.hrpulse.Controllers.ReportsControllers to  
javafx.fxml;  
    opens com.example.hrpulse.Services.Objects to org.hibernate.orm.core;  
    opens com.example.hrpulse.Services.CSV;  
    exports com.example.hrpulse.Services.Objects;  
    exports com.example.hrpulse.Services.Database to  
com.example.hrpulse.Services.CSV;  
    opens com.example.hrpulse.Services.Database;  
}
```

JasperReports jxml:

```
<?xml version="1.0" encoding="UTF-8"?>  
<!-- Created with Jaspersoft Studio version 6.20.6.final using  
JasperReports Library version 6.20.6--  
5c96b6aa8a39ac1dc6b6bea4b81168e16dd39231 -->  
<jasperReport xmlns="http://jasperreports.sourceforge.net/jasperreports"  
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
xsi:schemaLocation="http://jasperreports.sourceforge.net/jasperreports  
http://jasperreports.sourceforge.net/xsd/jasperreport.xsd" name="emo2"  
pageWidth="595" pageHeight="842" columnWidth="535" leftMargin="20"  
rightMargin="20" topMargin="20" bottomMargin="0" uuid="e9bb1a27-f975-4aa4-  
98f9-75524c5daafa">  
    <property name="com.jaspersoft.studio.data.sql.tables" value="" />  
    <property name="com.jaspersoft.studio.data.defaultdataadapter"  
value="New Data Adapter (5)"/>
```

```

<style name="Title" fontName="Arial" fontSize="26" isBold="true"/>
<style name="SubTitle" forecolor="#666666" fontName="Arial"
fontSize="18"/>
<style name="Column header" forecolor="#FFFFFF" fontName="Arial"
fontSize="12" isBold="true"/>
<style name="Detail" fontName="Arial" fontSize="12"/>
<queryString language="SQL">
<![CDATA[SELECT
    SUM(CAST(esd.total_work_hours AS DECIMAL(10, 2))) AS total_hours
FROM
    employeeeshiftdata esd]]>
</queryString>
<field name="total_work_hours" class="java.lang.String">
    <property name="com.jaspersoft.studio.field.name"
value="total_work_hours"/>
    <property name="com.jaspersoft.studio.field.label"
value="total_work_hours"/>
    <property name="com.jaspersoft.studio.field.tree.path"
value="employeeeshiftdata"/>
</field>
<field name="end_of_shift" class="java.lang.String">
    <property name="com.jaspersoft.studio.field.name"
value="end_of_shift"/>
    <property name="com.jaspersoft.studio.field.label"
value="end_of_shift"/>
    <property name="com.jaspersoft.studio.field.tree.path"
value="employeeeshiftdata"/>
</field>
<field name="start_of_shift" class="java.lang.String">
    <property name="com.jaspersoft.studio.field.name"
value="start_of_shift"/>
    <property name="com.jaspersoft.studio.field.label"
value="start_of_shift"/>
    <property name="com.jaspersoft.studio.field.tree.path"
value="employeeeshiftdata"/>
</field>
<field name="formatted_date" class="java.sql.Date">
    <property name="com.jaspersoft.studio.field.name"
value="formatted_date"/>
    <property name="com.jaspersoft.studio.field.label"
value="formatted_date"/>
    <property name="com.jaspersoft.studio.field.tree.path"
value="employeeeshiftdata"/>
</field>
<field name="last_name" class="java.lang.String">
    <property name="com.jaspersoft.studio.field.name"
value="last_name"/>
    <property name="com.jaspersoft.studio.field.label"
value="last_name"/>
    <property name="com.jaspersoft.studio.field.tree.path"
value="employees"/>
</field>
<field name="first_name" class="java.lang.String">
    <property name="com.jaspersoft.studio.field.name"
value="first_name"/>
    <property name="com.jaspersoft.studio.field.label"
value="first_name"/>
    <property name="com.jaspersoft.studio.field.tree.path"
value="employees"/>
</field>

```

```

<field name="employee_id" class="java.lang.Integer">
    <property name="com.jaspersoft.studio.field.name" value="id"/>
    <property name="com.jaspersoft.studio.field.label" value="employee_id"/>
</field>
<background>
    <band height="822" splitType="Stretch">
        <image vAlign="Bottom">
            <reportElement positionType="Float" mode="Transparent" x="-20" y="313" width="105" height="409" uuid="6be67a9c-1fb8-44ea-93c5-22c7de2c74ad"/>
                <imageExpression><! [CDATA[waves.jpg"]]></imageExpression>
            </image>
            <frame>
                <reportElement mode="Opaque" x="-20" y="722" width="105" height="100" forecolor="#666666" backcolor="#666666" uuid="58048a65-569d-454e-b6dd-f3bb46c94ee3"/>
            <box>
                <pen lineWidth="0.0"/>
                <topPen lineWidth="0.0"/>
                <leftPen lineWidth="0.0"/>
                <bottomPen lineWidth="0.0"/>
                <rightPen lineWidth="0.0"/>
            </box>
            <textField isBlankWhenNull="true">
                <reportElement style="Column header" x="0" y="0" width="104" height="73" forecolor="#000000" uuid="0f40ce36-78f2-49e5-adf5-8855d6516e74"/>
                    <textElement textAlignment="Center" verticalAlignment="Middle">
                        <font size="58" isBold="true"/>
                    </textElement>

<textFieldExpression><! [CDATA[$V{PAGE_NUMBER}]]></textFieldExpression>
            </textField>
            <textField evaluationTime="Report">
                <reportElement style="Column header" x="0" y="77" width="104" height="20" forecolor="#000000" uuid="e06a687c-0f3f-4451-a45e-f88c505ba589"/>
                    <box leftPadding="4" rightPadding="4"/>
                    <textElement textAlignment="Center">
                        <font size="16" isBold="false"/>
                    </textElement>

<textFieldExpression><! [CDATA[$V{PAGE_NUMBER}]]></textFieldExpression>
            </textField>
            <line>
                <reportElement mode="Transparent" x="42" y="73" width="20" height="1" uuid="ab96a4b0-8a0d-4ae2-8695-16868926ec2b"/>
            </line>
        </frame>
    </band>
</background>
<title>
    <band height="152" splitType="Stretch">
        <property name="com.jaspersoft.studio.layout"/>
        <staticText>
            <reportElement style="Title" x="0" y="0" width="555" height="115" uuid="524767a9-e7e5-4041-9b08-5a7210b38e22"/>

```

```

        <textElement textAlignment="Right" verticalAlignment="Middle">
            <font size="54" isBold="false"/>
        </textElement>
        <text><! [CDATA[נִזְבָּן וְנַעֲשֵׂה]]></text>
    </staticText>
</band>
</title>
<pageHeader>
    <band splitType="Stretch"/>
</pageHeader>
<columnHeader>
    <band height="25" splitType="Stretch">
        <frame>
            <reportElement mode="Opaque" x="10" y="5" width="544"
height="20" backcolor="#666666" uuid="eb0fefed-b830-492b-a0c9-
ec23411537a7"/>
            <staticText>
                <reportElement style="Column header" x="40" y="2"
width="66" height="16" uuid="6d0bdb9e-c985-42d7-82bb-6763df5c60d4">
                    <property
name="com.jaspersoft.studio.spreadsheet.connectionID" value="4358b8e6-8179-
4ce6-b48b-31c5ff8782da"/>
                </reportElement>
                <textElement textAlignment="Center"/>
                <text><! [CDATA[כ'ס נִזְבָּן]]></text>
            </staticText>
            <staticText>
                <reportElement style="Column header" x="106" y="0"
width="72" height="18" uuid="b522438d-8e47-461a-b24c-c4c6a7bb73f9">
                    <property
name="com.jaspersoft.studio.spreadsheet.connectionID" value="6068fc70-78cc-
4ed8-8053-2aadb0f57ec8"/>
                </reportElement>
                <textElement textAlignment="Center"/>
                <text><! [CDATA[סִינָת הַרְמֹן]]></text>
            </staticText>
            <staticText>
                <reportElement style="Column header" x="250" y="0"
width="97" height="18" uuid="93e6b141-0808-4d3f-813e-6bb12c2743ab">
                    <property
name="com.jaspersoft.studio.spreadsheet.connectionID" value="06e5f515-0112-
4a58-8b50-dffbb561b175"/>
                </reportElement>
                <textElement textAlignment="Center"/>
                <text><! [CDATA[לְתַאֲרִיךְ]]></text>
            </staticText>
            <staticText>
                <reportElement style="Column header" x="347" y="0"
width="65" height="18" uuid="9215506d-bcaf-4f4d-b899-b72456d8ef38">
                    <property
name="com.jaspersoft.studio.spreadsheet.connectionID" value="66763621-857f-
4618-9931-6b0390a9f2f3"/>
                </reportElement>
                <textElement textAlignment="Center"/>
                <text><! [CDATA[מִזְבָּחַ]]></text>
            </staticText>
            <staticText>
                <reportElement style="Column header" x="412" y="0"
width="65" height="18" uuid="29662c0d-8ea6-4d45-ba2a-2783efdd43ea">
                    <property
name="com.jaspersoft.studio.spreadsheet.connectionID" value="67e49161-3181-

```

```

4ae2-904e-06f3bf5adf1b"/>
    </reportElement>
    <textElement textAlignment="Center"/>
    <text><! [CDATA[מִשְׁרָטָה ]]></text>
</staticText>
<staticText>
    <reportElement style="Column header" x="477" y="0"
width="65" height="18" uuid="dd8f47ca-bd1d-4518-a3f2-58bb7e78c72b">
        <property
name="com.jaspersoft.studio.spreadsheet.connectionID" value="701d3572-8d48-
43f4-a9e7-4407dd628261"/>
            </reportElement>
            <textElement textAlignment="Center"/>
            <text><! [CDATA[n.t ]]></text>
        </staticText>
        <staticText>
            <reportElement style="Column header" x="178" y="0"
width="72" height="18" uuid="e196dedc-88a7-4ebe-beae-972cb21c0056">
                <property
name="com.jaspersoft.studio.spreadsheet.connectionID" value="6068fc70-78cc-
4ed8-8053-2aadb0f57ec8"/>
                    </reportElement>
                    <textElement textAlignment="Center"/>
                    <text><! [CDATA[תְּמִימָה תְּמִימָה ]]></text>
                </staticText>
            </frame>
        </band>
    </columnHeader>
    <detail>
        <band height="21" splitType="Stretch">
            <frame>
                <reportElement x="10" y="1" width="544" height="15"
uuid="5d75be3e-ea39-4b96-baf8-b49497315044"/>
                <textField>
                    <reportElement style="Detail" x="40" y="0" width="66"
height="15" uuid="dc2691cc-519b-4724-a80c-99365b21f34a">
                        <property
name="com.jaspersoft.studio.spreadsheet.connectionID" value="4358b8e6-8179-
4ce6-b48b-31c5ff8782da"/>
                            </reportElement>
                            <textElement textAlignment="Center"/>

<textFieldExpression><! [CDATA[$F{total_work_hours}]]></textFieldExpression>
                    </textField>
                    <textField>
                        <reportElement style="Detail" x="106" y="0" width="72"
height="15" uuid="7b53c877-6a6a-478c-92be-b8f2cae00566">
                            <property
name="com.jaspersoft.studio.spreadsheet.connectionID" value="6068fc70-78cc-
4ed8-8053-2aadb0f57ec8"/>
                                </reportElement>
                                <textElement textAlignment="Center"/>

<textFieldExpression><! [CDATA[$F{end_of_shift}]]></textFieldExpression>
                    </textField>
                    <textField>
                        <reportElement style="Detail" x="178" y="0" width="72"
height="15" uuid="3341329d-f870-4fb7-8d25-80d2e0373a86">
                            <property
name="com.jaspersoft.studio.spreadsheet.connectionID" value="4d8f3062-cac2-
4852-9341-fcb099be7f7"/>

```

```

        </reportElement>
        <textElement textAlignment="Center"/>

<textFieldExpression><![CDATA[$F{start_of_shift}]]></textFieldExpression>
        </textField>
        <textField>
            <reportElement style="Detail" x="347" y="0" width="65"
height="15" uuid="0207ff57-cf2d-4e1e-9acc-d7a9d5ec1078">
                <property
name="com.jaspersoft.studio.spreadsheet.connectionID" value="66763621-857f-
4618-9931-6b0390a9f2f3"/>
            </reportElement>
            <textElement textAlignment="Center"/>

<textFieldExpression><![CDATA[$F{last_name}]]></textFieldExpression>
        </textField>
        <textField>
            <reportElement style="Detail" x="412" y="0" width="65"
height="15" uuid="9c777dff-2c1f-4a29-976c-188f028c4e09">
                <property
name="com.jaspersoft.studio.spreadsheet.connectionID" value="67e49161-3181-
4ae2-904e-06f3bf5adf1b"/>
            </reportElement>
            <textElement textAlignment="Center"/>

<textFieldExpression><![CDATA[$F{first_name}]]></textFieldExpression>
        </textField>
        <textField>
            <reportElement style="Detail" x="477" y="0" width="65"
height="15" uuid="e509b394-3c98-4482-b84f-86983320a4bd">
                <property
name="com.jaspersoft.studio.spreadsheet.connectionID" value="701d3572-8d48-
43f4-a9e7-4407dd628261"/>
            </reportElement>
            <textElement textAlignment="Center"/>

<textFieldExpression><![CDATA[$F{employee_id}]]></textFieldExpression>
        </textField>
        <textField>
            <reportElement style="Detail" x="250" y="0" width="97"
height="15" uuid="1462f5d1-8797-45a4-8762-f8c9e1d37599">
                <property
name="com.jaspersoft.studio.spreadsheet.connectionID" value="4d8f3062-cac2-
4852-9341-fcb099be7f7"/>
            </reportElement>
            <textElement textAlignment="Center"/>
            <textFieldExpression><![CDATA[new
SimpleDateFormat("dd/MM/yyyy").format($F{formatted_date})]]></textFieldExpression>
        </textField>

        </frame>
        <line>
            <reportElement positionType="FixRelativeToBottom" x="10"
y="16" width="545" height="1" uuid="2b97af27-2a1e-491f-a0f4-b2549a3510ee"/>
            <graphicElement>
                <pen lineWidth="0.5" lineColor="#999999"/>
            </graphicElement>
        </line>
    </band>
```

```

</detail>
<columnFooter>
    <band splitType="Stretch"/>
</columnFooter>
<pageFooter>
    <band height="100" splitType="Stretch">
        <textField pattern="EEEEEE dd MMMMM yyyy">
            <reportElement style="Column header" x="410" y="30"
width="131" height="20" forecolor="#000000" uuid="f8570c2f-4404-4725-885a-
4eb3c26abb4a"/>
            <textElement textAlign="Right">
                <font size="12" isBold="false"/>
            </textElement>
            <textFieldExpression><![CDATA[new
java.util.Date()]]></textFieldExpression>
        </textField>
    </band>
</pageFooter>
<summary>
    <band splitType="Stretch"/>
</summary>
</jasperReport>

```

EmployeeReport jrxml:

```

<?xml version="1.0" encoding="UTF-8"?>
<!-- Created with Jaspersoft Studio version 6.20.6.final using
JasperReports Library version 6.20.6-
5c96b6aa8a39ac1dc6b6bea4b81168e16dd39231 -->
<jasperReport xmlns="http://jasperreports.sourceforge.net/jasperreports"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://jasperreports.sourceforge.net/jasperreports
http://jasperreports.sourceforge.net/xsd/jasperreport.xsd"
name="Waves_Landscape" pageWidth="842" pageHeight="595"
orientation="Landscape" columnWidth="802" leftMargin="20" rightMargin="20"
topMargin="20" bottomMargin="0" uuid="e115bc59-fd0f-446b-a585-
04725e27d7f0">
    <property name="com.jaspersoft.studio.data.sql.tables" value="" />
    <property name="com.jaspersoft.studio.data.defaultdataadapter"
value="New Data Adapter"/>
    <style name="Title" fontName="Arial" fontSize="26" isBold="true"/>
    <style name="SubTitle" forecolor="#666666" fontName="Arial"
fontSize="18"/>
    <style name="Column header" forecolor="#FFFFFF" fontName="Arial"
fontSize="12" isBold="true"/>
    <style name="Detail" fontName="Arial" fontSize="12"/>
    <queryString language="SQL">
        <![CDATA[select *from employees]]>
    </queryString>
    <field name="first_name" class="java.lang.String">
        <property name="com.jaspersoft.studio.field.name"
value="first_name"/>
        <property name="com.jaspersoft.studio.field.label"
value="first_name"/>
        <property name="com.jaspersoft.studio.field.tree.path"
value="employees"/>
    </field>
    <field name="last_name" class="java.lang.String">
        <property name="com.jaspersoft.studio.field.name"
value="last_name"/>

```

```

<property name="com.jaspersoft.studio.field.label"
value="last_name"/>
    <property name="com.jaspersoft.studio.field.tree.path"
value="employees"/>
</field>
<field name="phone_number" class="java.lang.String">
    <property name="com.jaspersoft.studio.field.name"
value="phone_number"/>
        <property name="com.jaspersoft.studio.field.label"
value="phone_number"/>
            <property name="com.jaspersoft.studio.field.tree.path"
value="employees"/>
</field>
<field name="email" class="java.lang.String">
    <property name="com.jaspersoft.studio.field.name" value="email"/>
    <property name="com.jaspersoft.studio.field.label" value="email"/>
        <property name="com.jaspersoft.studio.field.tree.path"
value="employees"/>
</field>
<field name="department" class="java.lang.String">
    <property name="com.jaspersoft.studio.field.name"
value="department"/>
        <property name="com.jaspersoft.studio.field.label"
value="department"/>
            <property name="com.jaspersoft.studio.field.tree.path"
value="employees"/>
</field>
<field name="salaryPerMonth" class="java.lang.Double">
    <property name="com.jaspersoft.studio.field.name"
value="salaryPerMonth"/>
        <property name="com.jaspersoft.studio.field.label"
value="salaryPerMonth"/>
            <property name="com.jaspersoft.studio.field.tree.path"
value="employees"/>
</field>
<background>
    <band height="575" splitType="Stretch">
        <image vAlign="Bottom">
            <reportElement positionType="Float" x="-20" y="66"
width="105" height="409" uuid="64ca4020-b604-4de9-9ed1-921b74cf593"/>
                <imageExpression><![CDATA[waves.jpg"]]></imageExpression>
            </image>
        <frame>
            <reportElement mode="Opaque" x="-20" y="475" width="105"
height="100" backcolor="#666666" uuid="6eb7cb61-0dd7-47a8-8475-
5fa6c3a348dc"/>
                <textField>
                    <reportElement style="Column header" x="0" y="0"
width="104" height="73" forecolor="#000000" uuid="db8457c8-99e9-434f-b439-
2d400dff2bcd"/>
                        <textElement textAlignment="Center"
verticalAlignment="Middle">
                            <font size="58" isBold="true"/>
                        </textElement>

<textFieldExpression><![CDATA[$V{PAGE_NUMBER}"]]></textFieldExpression>
                    </textField>
                    <textField evaluationTime="Report">
                        <reportElement style="Column header" x="0" y="77"
width="104" height="20" forecolor="#000000" uuid="e20c8a65-aded-41dc-a107-
108a9159af99"/>

```

```

<box leftPadding="4" rightPadding="4"/>
<textElement textAlignment="Center">
    <font size="16" isBold="false"/>
</textElement>

<textFieldExpression><![CDATA[$V{PAGE_NUMBER}]]></textFieldExpression>
</textField>
<line>
    <reportElement mode="Transparent" x="42" y="73"
width="20" height="1" backcolor="#FFFFFF" uuid="db6d7ead-4383-4f8c-81cf-
1e3064eb6924"/>
    </line>
</frame>
</band>
</background>
<title>
    <band height="104" splitType="Stretch">
        <staticText>
            <reportElement style="Title" x="0" y="0" width="802"
height="67" uuid="1dcadd54-7d50-4470-85ed-189e4d910b25"/>
            <textElement textAlignment="Right"
verticalAlignment="Middle">
                <font size="54" isBold="false"/>
            </textElement>
            <text><![CDATA[HR Pulse]]></text>
        </staticText>
        <staticText>
            <reportElement style="SubTitle" x="232" y="67" width="568"
height="37" uuid="862cb50c-5a5e-4906-a495-226e25115fb6"/>
            <textElement textAlignment="Right">
                <font size="26"/>
            </textElement>
            <text><![CDATA[חברה עובדי]]></text>
        </staticText>
    </band>
</title>
<pageHeader>
    <band splitType="Stretch"/>
</pageHeader>
<columnHeader>
    <band height="25" splitType="Stretch">
        <frame>
            <reportElement mode="Opaque" x="97" y="5" width="705"
height="20" backcolor="#666666" uuid="35d90a9b-bb7e-4ec4-bfa2-
68be1967d6b2"/>
            <staticText>
                <reportElement style="Column header" x="0" y="0"
width="117" height="15" uuid="860b6bd6-deb4-4485-9d86-9b47e34b2204">
                    <property
name="com.jaspersoft.studio.spreadsheet.connectionID" value="fa5abc9a-cb07-
4771-96da-cbafaf4a7acd"/>
                </reportElement>
                <textElement textAlignment="Center"/>
                <text><![CDATA[נץ]]></text>
            </staticText>
            <staticText>
                <reportElement style="Column header" x="117" y="0"
width="117" height="15" uuid="021841c9-8c30-4c74-8962-0ba04ab4e467">
                    <property
name="com.jaspersoft.studio.spreadsheet.connectionID" value="4594881c-5fdd-
4e2d-9d18-467d85e0dfdf"/>

```

```

        </reportElement>
        <textElement textAlignment="Center"/>
        <text><![CDATA[הספורה בולט]]></text>
    </staticText>
    <staticText>
        <reportElement style="Column header" x="234" y="0"
width="117" height="15" uuid="65bb8b1a-1b43-4871-b34a-575e436be313">
            <property
name="com.jaspersoft.studio.spreadsheet.connectionID" value="ea9c7671-b003-
46b2-941b-cedc947b6691"/>
        </reportElement>
        <textElement textAlignment="Center"/>
        <text><![CDATA[?]]></text>
    </staticText>
    <staticText>
        <reportElement style="Column header" x="351" y="0"
width="117" height="15" uuid="515e0067-41f7-4ee7-907c-1e8336deaad6">
            <property
name="com.jaspersoft.studio.spreadsheet.connectionID" value="55165216-7f24-
4a6e-9d7a-de251ec4f9b2"/>
        </reportElement>
        <textElement textAlignment="Center"/>
        <text><![CDATA[ניד הספורה]]></text>
    </staticText>
    <staticText>
        <reportElement style="Column header" x="450" y="0"
width="117" height="15" uuid="682957bb-4123-4db2-9172-068fa82f412e">
            <property
name="com.jaspersoft.studio.spreadsheet.connectionID" value="fa55e06f-715e-
468b-9713-be58e89ebcef"/>
        </reportElement>
        <textElement textAlignment="Center"/>
        <text><![CDATA[הספורה]]></text>
    </staticText>
    <staticText>
        <reportElement style="Column header" x="581" y="3"
width="117" height="15" uuid="04748cec-b14c-4275-b8d8-93ded7bbd5ba">
            <property
name="com.jaspersoft.studio.spreadsheet.connectionID" value="d85a94f3-967a-
4f78-907a-36885d495a60"/>
        </reportElement>
        <textElement textAlignment="Center"/>
        <text><![CDATA[פרטי ]]></text>
    </staticText>
</frame>
</band>
</columnHeader>
<detail>
    <band height="21" splitType="Stretch">
        <frame>
            <reportElement x="97" y="1" width="705" height="15"
uuid="89a3da05-fdde-4ffe-965f-cc476f72e3ab"/>
            <textField>
                <reportElement style="Detail" x="587" y="2" width="117"
height="15" uuid="0f39eace-3715-4652-84b5-be075b43dec0">
                    <property
name="com.jaspersoft.studio.spreadsheet.connectionID" value="fa5abc9a-cb07-
4771-96da-cbafaf4a7acd"/>
                </reportElement>

<textFieldExpression><![CDATA[$F{first_name}]]></textFieldExpression>

```

```
</textField>
<textField>
    <reportElement style="Detail" x="489" y="0" width="117"
height="15" uuid="c7bd7e26-ce21-495e-bf43-4d566c6a3383">
        <property
name="com.jaspersoft.studio.spreadsheet.connectionID" value="4594881c-5fdd-
4e2d-9d18-467d85e0dfdf"/>
    </reportElement>

<textFieldExpression><![CDATA[$F{last_name}]]></textFieldExpression>
    </textField>
    <textField>
        <reportElement style="Detail" x="357" y="0" width="117"
height="15" uuid="9e52cf6a-cdca-48c0-8221-9b69282ea080">
            <property
name="com.jaspersoft.studio.spreadsheet.connectionID" value="ea9c7671-b003-
46b2-941b-cedc947b6691"/>
        </reportElement>

<textFieldExpression><![CDATA[$F{phone_number}]]></textFieldExpression>
    </textField>
    <textField>
        <reportElement style="Detail" x="253" y="0" width="117"
height="15" uuid="b7a114e3-ffdd-4536-9215-c533cela055f">
            <property
name="com.jaspersoft.studio.spreadsheet.connectionID" value="55165216-7f24-
4a6e-9d7a-de251ec4f9b2"/>
        </reportElement>

<textFieldExpression><![CDATA[$F{email}]]></textFieldExpression>
    </textField>
    <textField>
        <reportElement style="Detail" x="3" y="0" width="117"
height="15" uuid="d01abe93-eca0-4ec2-a8d1-881f50633efb">
            <property
name="com.jaspersoft.studio.spreadsheet.connectionID" value="fa55e06f-715e-
468b-9713-be58e89ebcef"/>
        </reportElement>

<textFieldExpression><![CDATA[$F{department}]]></textFieldExpression>
    </textField>
    <textField>
        <reportElement style="Detail" x="122" y="0" width="117"
height="15" uuid="9edb223e-c594-44ca-a44c-c48907b90559">
            <property
name="com.jaspersoft.studio.spreadsheet.connectionID" value="d85a94f3-967a-
4f78-907a-36885d495a60"/>
        </reportElement>

<textFieldExpression><![CDATA[$F{salaryPerMonth}]]></textFieldExpression>
    </textField>
    </frame>
    <line>
        <reportElement positionType="FixRelativeToBottom" x="97"
y="16" width="705" height="1" uuid="e5f21b91-7f24-498e-97a1-0e93db9225a4"/>
        <graphicElement>
            <pen lineWidth="0.5" lineColor="#999999"/>
        </graphicElement>
    </line>
    </band>
</detail>
```

```

<columnFooter>
    <band splitType="Stretch"/>
</columnFooter>
<pageFooter>
    <band height="139" splitType="Stretch">
        <textField pattern="EEEEEE dd MMMMM yyyy">
            <reportElement style="Column header" x="671" y="0"
width="131" height="20" forecolor="#000000" uuid="95484ca0-40b4-4752-95d8-
1afbc33c820b"/>
            <textElement textAlignment="Right">
                <font size="12" isBold="false"/>
            </textElement>
            <textFieldExpression><![CDATA[new
java.util.Date()]]></textFieldExpression>
        </textField>
        <staticText>
            <reportElement x="96" y="0" width="267" height="20"
uuid="0ec52099-dcd7-42f4-8baa-f3874f05208a"/>
            <textElement>
                <font size="12"/>
            </textElement>
            <text><![CDATA[HR PULSE Report]]></text>
        </staticText>
    </band>
</pageFooter>
<summary>
    <band splitType="Stretch"/>
</summary>
</jasperReport>

```

Employeeeshift report jrxml:

```

Employeeeshift report jrxml:
<?xml version="1.0" encoding="UTF-8"?>
<!-- Created with Jaspersoft Studio version 6.20.6.final using
JasperReports Library version 6.20.6-
5c96b6aa8a39ac1dc6b6bea4b81168e16dd39231 --&gt;
&lt;jasperReport xmlns="http://jasperreports.sourceforge.net/jasperreports"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://jasperreports.sourceforge.net/jasperreports
http://jasperreports.sourceforge.net/xsd/jasperreport.xsd" name="Waves"
pageWidth="595" pageHeight="842" columnWidth="535" leftMargin="20"
rightMargin="20" topMargin="20" bottomMargin="0" uuid="e9bb1a27-f975-4aa4-
98f9-75524c5daafa"&gt;
    &lt;property name="com.jaspersoft.studio.data.sql.tables" value="" /&gt;
    &lt;property name="com.jaspersoft.studio.data.defaultdataadapter"
value="New Data Adapter (3)" /&gt;
    &lt;style name="Title" fontName="Arial" fontSize="26" isBold="true" /&gt;
    &lt;style name="SubTitle" forecolor="#666666" fontName="Arial"
fontSize="18" /&gt;
    &lt;style name="Column header" forecolor="#FFFFFF" fontName="Arial"
fontSize="12" isBold="true" /&gt;
    &lt;style name="Detail" fontName="Arial" fontSize="12" /&gt;
    &lt;queryString language="SQL"&gt;
        &lt;![CDATA[select * From employeeeshiftdata]]&gt;
    &lt;/queryString&gt;
    &lt;field name="employee_id" class="java.lang.String"&gt;
        &lt;property name="com.jaspersoft.studio.field.name"
value="employee_id"/&gt;
        &lt;property name="com.jaspersoft.studio.field.label"
</pre>

```

```

value="employee_id"/>
    <property name="com.jaspersoft.studio.field.tree.path"
value="employeeshiftdata"/>
</field>
<field name="start_of_shift" class="java.lang.String">
    <property name="com.jaspersoft.studio.field.name"
value="start_of_shift"/>
    <property name="com.jaspersoft.studio.field.label"
value="start_of_shift"/>
    <property name="com.jaspersoft.studio.field.tree.path"
value="employeeshiftdata"/>
</field>
<field name="end_of_shift" class="java.lang.String">
    <property name="com.jaspersoft.studio.field.name"
value="end_of_shift"/>
    <property name="com.jaspersoft.studio.field.label"
value="end_of_shift"/>
    <property name="com.jaspersoft.studio.field.tree.path"
value="employeeshiftdata"/>
</field>
<field name="total_work_hours" class="java.lang.String">
    <property name="com.jaspersoft.studio.field.name"
value="total_work_hours"/>
    <property name="com.jaspersoft.studio.field.label"
value="total_work_hours"/>
    <property name="com.jaspersoft.studio.field.tree.path"
value="employeeshiftdata"/>
</field>
<group name="Group1">
    <groupExpression><![CDATA[$F{employee_id}]]></groupExpression>
    <groupHeader>
        <band height="40">
            <textField>
                <reportElement style="SubTitle" x="143" y="13" width="410"
height="24" forecolor="#000000" uuid="91d5865d-2791-4d53-8ab5-
f5bedeeaa756b"/>
                    <textElement>
                        <font isBold="true"/>
                    </textElement>

<textFieldExpression><![CDATA[$F{employee_id}]]></textFieldExpression>
            </textField>
            <rectangle>
                <reportElement mode="Opaque" x="97" y="13" width="36"
height="24" forecolor="#CCCCCC" backcolor="#CCCCCC" uuid="a60f080a-41ef-
421c-93cf-34257c2e2b0a"/>
            </rectangle>
        </band>
    </groupHeader>
    <groupFooter>
        <band height="30">
            <frame>
                <reportElement mode="Opaque" x="405" y="0" width="150"
height="30" forecolor="#CCCCCC" backcolor="#CCCCCC" uuid="8dc6af22-fec6-
467d-8eaf-f215f6a6684e"/>
            </frame>
        </band>
    </groupFooter>
</group>
<background>
    <band height="822" splitType="Stretch">

```

```

<image vAlign="Bottom">
    <reportElement positionType="Float" mode="Transparent" x="-20"
y="313" width="105" height="409" uuid="6be67a9c-1fb8-44ea-93c5-
22c7de2c74ad"/>
    <imageExpression><![CDATA[waves.jpg]]></imageExpression>
</image>
<frame>
    <reportElement mode="Opaque" x="-20" y="722" width="105"
height="100" forecolor="#666666" backcolor="#666666" uuid="58048a65-569d-
454e-b6dd-f3bb46c94ee3"/>
    <box>
        <pen lineWidth="0.0"/>
        <topPen lineWidth="0.0"/>
        <leftPen lineWidth="0.0"/>
        <bottomPen lineWidth="0.0"/>
        <rightPen lineWidth="0.0"/>
    </box>
    <textField isBlankWhenNull="true">
        <reportElement style="Column header" x="0" y="0"
width="104" height="73" forecolor="#000000" uuid="0f40ce36-78f2-49e5-adf5-
8855d6516e74"/>
        <textElement textAlignment="Center"
verticalAlignment="Middle">
            <font size="58" isBold="true"/>
        </textElement>

<textFieldExpression><![CDATA[$V{PAGE_NUMBER}]]></textFieldExpression>
        </textField>
        <textField evaluationTime="Report">
            <reportElement style="Column header" x="0" y="77"
width="104" height="20" forecolor="#000000" uuid="e06a687c-0f3f-4451-a45e-
f88c505ba589"/>
            <box leftPadding="4" rightPadding="4"/>
            <textElement textAlignment="Center">
                <font size="16" isBold="false"/>
            </textElement>

<textFieldExpression><![CDATA[$V{PAGE_NUMBER}]]></textFieldExpression>
        </textField>
        <line>
            <reportElement mode="Transparent" x="42" y="73" width="20"
height="1" uuid="ab96a4b0-8a0d-4ae2-8695-16868926ec2b"/>
        </line>
    </frame>
</band>
</background>
<title>
    <band height="152" splitType="Stretch">
        <staticText>
            <reportElement style="Title" x="0" y="0" width="555"
height="115" uuid="524767a9-e7e5-4041-9b08-5a7210b38e22"/>
            <textElement textAlignment="Right" verticalAlignment="Middle">
                <font size="54" isBold="false"/>
            </textElement>
            <text><![CDATA[WAVES TITLE]]></text>
        </staticText>
        <staticText>
            <reportElement style="SubTitle" x="234" y="115" width="321"
height="37" uuid="bd605e96-6228-492c-9dd6-f33258081a5a"/>
            <textElement textAlignment="Right">
                <font size="26"/>
            </textElement>
        </staticText>
    </band>
</title>

```

```

        </textElement>
        <text><![CDATA[Waves SubTitle]]></text>
    </staticText>
</band>
</title>
<pageHeader>
    <band splitType="Stretch"/>
</pageHeader>
<columnHeader>
    <band height="25" splitType="Stretch">
        <frame>
            <reportElement mode="Opaque" x="97" y="5" width="458"
height="20" backcolor="#666666" uuid="eb0fefed-b830-492b-a0c9-
ec23411537a7"/>
            <staticText>
                <reportElement style="Column header" x="0" y="0"
width="152" height="15" uuid="a4c305c3-38bd-4b2a-89ab-086107ba861e">
                    <property
name="com.jaspersoft.studio.spreadsheet.connectionID" value="2dbf6de7-a52d-
4e9d-832b-9adabbed2ba8"/>
                </reportElement>
                <text><![CDATA[start_of_shift]]></text>
            </staticText>
            <staticText>
                <reportElement style="Column header" x="152" y="0"
width="152" height="15" uuid="607cf4fe-6804-4692-a9e9-f4d01ced796a">
                    <property
name="com.jaspersoft.studio.spreadsheet.connectionID" value="b838f657-f904-
4b56-8157-d045ba321063"/>
                </reportElement>
                <text><![CDATA[end_of_shift]]></text>
            </staticText>
            <staticText>
                <reportElement style="Column header" x="304" y="0"
width="152" height="15" uuid="bfaabe01-0cec-4d69-94d2-4e4a0dca1636">
                    <property
name="com.jaspersoft.studio.spreadsheet.connectionID" value="349a9256-376e-
4105-8448-1e9d8edf99aa"/>
                </reportElement>
                <text><![CDATA[total_work_hours]]></text>
            </staticText>
        </frame>
    </band>
</columnHeader>
<detail>
    <band height="21" splitType="Stretch">
        <frame>
            <reportElement x="97" y="1" width="458" height="15"
uuid="5d75be3e-ea39-4b96-baf8-b49497315044"/>
            <textField>
                <reportElement style="Detail" x="0" y="0" width="152"
height="15" uuid="02cbfa91-b51d-48f9-bb33-c812076deb7">
                    <property
name="com.jaspersoft.studio.spreadsheet.connectionID" value="2dbf6de7-a52d-
4e9d-832b-9adabbed2ba8"/>
                </reportElement>

<textFieldExpression><![CDATA[$F{start_of_shift}]]></textFieldExpression>
            </textField>
            <textField>
                <reportElement style="Detail" x="152" y="0" width="152"

```

```
height="15" uuid="bcc319bf-1686-4269-82f2-e9f0d8745af2">
    <property
name="com.jaspersoft.studio.spreadsheet.connectionID" value="b838f657-f904-
4b56-8157-d045ba321063"/>
    </reportElement>

<textFieldExpression><! [CDATA[$F{end_of_shift}] ]></textFieldExpression>
    </textField>
    <textField>
        <reportElement style="Detail" x="304" y="0" width="152"
height="15" uuid="27104d6f-f9d2-4b56-ab1c-e45dd1fada47">
            <property
name="com.jaspersoft.studio.spreadsheet.connectionID" value="349a9256-376e-
4105-8448-1e9d8edf99aa"/>
        </reportElement>

<textFieldExpression><! [CDATA[$F{total_work_hours}] ]></textFieldExpression>
    </textField>
    </frame>
    <line>
        <reportElement positionType="FixRelativeToBottom" x="97"
y="16" width="458" height="1" uuid="2b97af27-2a1e-491f-a0f4-b2549a3510ee"/>
        <graphicElement>
            <pen lineWidth="0.5" lineColor="#999999"/>
        </graphicElement>
    </line>
    </band>
</detail>
<columnFooter>
    <band splitType="Stretch"/>
</columnFooter>
<pageFooter>
    <band height="100" splitType="Stretch">
        <textField pattern="EEEEEE dd MMMMM yyyy">
            <reportElement style="Column header" x="424" y="0" width="131"
height="20" forecolor="#000000" uuid="f8570c2f-4404-4725-885a-
4eb3c26abb4a">
                <textElement textAlignment="Right">
                    <font size="12" isBold="false"/>
                </textElement>
                <textFieldExpression><! [CDATA[new
java.util.Date()] ]></textFieldExpression>
                </textField>
                <staticText>
                    <reportElement style="Column header" x="96" y="0" width="267"
height="20" forecolor="#000000" uuid="fce324aa-31f5-453a-a957-
28bc6f740c75"/>
                    <textElement>
                        <font size="12" isBold="false"/>
                    </textElement>
                    <text><! [CDATA[Waves Title Report] ]></text>
                </staticText>
            </band>
</pageFooter>
<summary>
    <band splitType="Stretch"/>
</summary>
</jasperReport>
```

Dependencies POM file - used Maven:

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
          xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
https://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>

    <groupId>com.example</groupId>
    <artifactId>HR-pulse</artifactId>
    <version>1.0-SNAPSHOT</version>
    <name>HR-pulse</name>

    <properties>
        <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
        <junit.version>5.9.2</junit.version>
    </properties>

    <dependencies>

        <dependency>
            <groupId>org.openjfx</groupId>
            <artifactId>javafx-controls</artifactId>
            <version>20.0.1</version>
        </dependency>
        <!-- OpenCSV -->
        <dependency>
            <groupId>com.opencsv</groupId>
            <artifactId>opencsv</artifactId>
            <version>4.1</version>
        </dependency>
        <!-- Hibernate -->
        <dependency>
            <groupId>org.hibernate</groupId>
            <artifactId>hibernate-core</artifactId>
            <version>5.5.7.Final</version>
        </dependency>
        <!-- My SqlConnector-->
        <!-- https://mvnrepository.com/artifact/com.mysql/mysql-connector-j
-->
        <dependency>
            <groupId>com.mysql</groupId>
            <artifactId>mysql-connector-j</artifactId>
            <version>8.1.0</version>
        </dependency>
        <!--java mail -->
        <dependency>
            <groupId>com.sun.mail</groupId>
            <artifactId>javax.mail</artifactId>
            <version>1.6.2</version>
        </dependency>

        <dependency>
            <groupId>org.openjfx</groupId>
            <artifactId>javafx-fxml</artifactId>
            <version>19</version>
        </dependency>
    </dependencies>

```

```

<dependency>
    <groupId>org.junit.jupiter</groupId>
    <artifactId>junit-jupiter-api</artifactId>
    <version>${junit.version}</version>
    <scope>test</scope>
</dependency>

<dependency>
    <groupId>net.sf.jasperreports</groupId>
    <artifactId>jasperreports</artifactId>
    <version>6.17.0</version>
</dependency>
<dependency>
    <groupId>org.junit.jupiter</groupId>
    <artifactId>junit-jupiter-engine</artifactId>
    <version>${junit.version}</version>
    <scope>test</scope>
</dependency>

</dependencies>

<build>
    <plugins>
        <plugin>
            <groupId>org.apache.maven.plugins</groupId>
            <artifactId>maven-compiler-plugin</artifactId>
            <version>3.8.1</version>
            <configuration>
                <source>9</source>
                <target>9</target>
            </configuration>
        </plugin>
        <plugin>
            <groupId>org.openjfx</groupId>
            <artifactId>javafx-maven-plugin</artifactId>
            <version>0.0.8</version>
            <executions>
                <execution>
                    <!-- Default configuration for running with: mvn
clean javafx:run -->
                    <id>default-cli</id>
                    <configuration>
<mainClass>com.example.hrpulse/com.example.hrpulse.DatabaseManager</mainClass>
                    <launcher>app</launcher>
                    <jlinkZipName>app</jlinkZipName>
                    <jlinkImageName>app</jlinkImageName>
                    <noManPages>true</noManPages>
                    <stripDebug>true</stripDebug>
                    <noHeaderFiles>true</noHeaderFiles>
                </configuration>
                </execution>
            </executions>
        </plugin>
    </plugins>
</build>
</project>

```

Login View xml:

```
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.geometry.Insets?>
<?import javafx.scene.control.Button?>
<?import javafx.scene.control.Label?>
<?import javafx.scene.control.PasswordField?>
<?import javafx.scene.control.TextField?>
<?import javafx.scene.effect.DropShadow?>
<?import javafx.scene.effect.Glow?>
<?import javafx.scene.effect.InnerShadow?>
<?import javafx.scene.image.Image?>
<?import javafx.scene.image.ImageView?>
<?import javafx.scene.layout.AnchorPane?>
<?import javafx.scene.layout.BorderPane?>
<?import javafx.scene.shape.Line?>
<?import javafx.scene.text.Font?>

<BorderPane xmlns="http://javafx.com/javafx/19"
xmlns:fx="http://javafx.com/fxml/1"
fx:controller="com.example.hrpulse.Controllers.loginController">
    <left>
        <AnchorPane style="-fx-background-color: #263f73;" BorderPane.alignment="CENTER">
            <children>
                <Label alignment="CENTER" layoutX="20.0" layoutY="61.0" prefHeight="24.0" prefWidth="182.0" text="HR PULSE" textFill="WHITE">
                    <font>
                        <Font name="Courier New Bold" size="29.0" />
                    </font>
                </Label>
                <Line endX="81.0" layoutX="124.0" layoutY="96.0" startX="-100.0" stroke="#f5f3f3" strokeWidth="2.0" />
                <Label layoutX="44.0" layoutY="110.0" text="חתפיה של HR PULSE" textFill="#fcfafa" />
                <Label layoutX="31.0" layoutY="398.0" text="Copyright © 2023 A&S Group" textFill="#f8f8f8" />
                <ImageView fitHeight="150.0" fitWidth="200.0" layoutX="15.0" layoutY="161.0" pickOnBounds="true" preserveRatio="true">
                    <image>
                        <Image url="@../../../../Images/HR_logo.jpg" />
                    </image>
                </ImageView>
            </children>
            <BorderPane.margin>
                <Insets bottom="10.0" left="10.0" right="10.0" top="10.0" />
            </BorderPane.margin>
        </AnchorPane>
    </left>
    <right>
        <AnchorPane prefHeight="465.0" prefWidth="348.0" BorderPane.alignment="CENTER">
            <children>
                <Label alignment="CENTER" layoutX="71.0" layoutY="16.0" prefHeight="30.0" prefWidth="195.0" style="-fx-background-color: white;" text="הכנסו למשתמש" textAlignment="CENTER">
                    <effect>
                        <InnerShadow />
                    </effect>
                    <font>
```

```

        <Font name="Comic Sans MS Bold Italic" size="14.0" />
    </font>
</Label>
<Label alignment="CENTER" layoutX="209.0" layoutY="123.0"
prefHeight="43.0" prefWidth="114.0" text="ת.ת :" textAlignment="CENTER">
    <font>
        <Font name="Comic Sans MS" size="14.0" />
    </font>
    <effect>
        <Glow />
    </effect>
</Label>
<TextField fx:id="tf_UserName" alignment="CENTER"
layoutX="73.0" layoutY="132.0" promptText="ת.ת טבון" />
<Label alignment="CENTER" layoutX="241.0" layoutY="186.0"
text="סיסמה :">
    <font>
        <Font name="Comic Sans MS" size="14.0" />
    </font>
    <effect>
        <Glow />
    </effect>
</Label>
<PasswordField fx:id="tf_Password" alignment="CENTER"
layoutX="73.0" layoutY="184.0" promptText="סיסמה" />
<Button layoutX="42.0" layoutY="324.0" mnemonicParsing="false"
onAction="#userLogin" prefHeight="39.0" prefWidth="238.0" style="-fx-
background-color: #1d28bf;" text="הניכר" textAlignment="CENTER"
textFill="#f5f2f2">
    <font>
        <Font name="System Bold Italic" size="18.0" />
    </font>
    <effect>
        <InnerShadow>
            <input>
                <DropShadow />
            </input>
        </InnerShadow>
    </effect>
</Button>
<Button fx:id="exitButton" layoutX="42.0" layoutY="394.0"
mnemonicParsing="false" onAction="#ExitButtonClicked" prefHeight="39.0"
prefWidth="239.0" style="-fx-background-color: #1d28bf;" text="خروج"
textFill="#f4f4f8">
    <font>
        <Font name="System Bold Italic" size="18.0" />
    </font>
    <effect>
        <InnerShadow />
    </effect>
</Button>
<Label fx:id="wrongLogin" contentDisplay="CENTER"
layoutX="112.0" layoutY="244.0" textFill="#e40808" />
</children>
<padding>
    <Insets bottom="10.0" left="10.0" right="10.0" top="10.0" />
</padding>
<BorderPane.margin>
    <Insets bottom="10.0" left="10.0" right="10.0" top="10.0" />
</BorderPane.margin>
</AnchorPane>

```

```
        </right>  
</BorderPane>
```



DepartmentView FXMLs:

createNewDepartment_view.fxml

```
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.geometry.Insets?>
<?import javafx.scene.control.Button?>
<?import javafx.scene.control.ButtonBar?>
<?import javafx.scene.control.Label?>
<?import javafx.scene.control.TextField?>
<?import javafx.scene.effect.Glow?>
<?import javafx.scene.effect.InnerShadow?>
<?import javafx.scene.effect.MotionBlur?>
<?import javafx.scene.image.Image?>
<?import javafx.scene.image.ImageView?>
<?import javafx.scene.layout.AnchorPane?>
<?import javafx.scene.layout.BorderPane?>
<?import javafx.scene.layout.ColumnConstraints?>
<?import javafx.scene.layout.GridPane?>
<?import javafx.scene.layout.RowConstraints?>
<?import javafx.scene.text.Font?>

<AnchorPane prefHeight="433.0" prefWidth="615.0" style="-fx-background-color: grey;" xmlns="http://javafx.com/javafx/19"
xmlns:fx="http://javafx.com/fxml/1"
fx:controller="com.example.hrpulse.Controllers.DepartmentController.CreateNewDepartmentController">
    <children>
        <AnchorPane layoutX="60.0" layoutY="56.0" prefHeight="333.0" prefWidth="517.0" style="-fx-background-color: white;" AnchorPane.bottomAnchor="20.0" AnchorPane.leftAnchor="20.0" AnchorPane.rightAnchor="20.0" AnchorPane.topAnchor="20.0">
            <children>
                <BorderPane layoutX="151.0" layoutY="65.0" prefHeight="200.0" prefWidth="200.0" AnchorPane.bottomAnchor="20.0" AnchorPane.leftAnchor="20.0" AnchorPane.rightAnchor="20.0" AnchorPane.topAnchor="20.0">
                    <top>
                        <Label text="הווסף מילקה חדשה" textFill="#757a9c" BorderPane.alignment="CENTER">
                            <font>
                                <Font name="System Bold Italic" size="18.0" />
                            </font>
                            <effect>
                                <Glow>
                                    <input>
                                        <InnerShadow />
                                    </input>
                                </Glow>
                            </effect>
                        </Label>
                    </top>
                    <center>
                        <Form fx:id="newDepartmentForm" BorderPane.alignment="CENTER">
                            <GridPane columns="2" BorderPane.alignment="CENTER">
                                <Label text="שם המילקה" GridPane.columnIndex="0" GridPane.rowIndex="1" BorderPane.alignment="CENTER"/>
                                <TextField fx:id="nameField" GridPane.columnIndex="1" GridPane.rowIndex="1" BorderPane.alignment="CENTER"/>
                                <Label text="כתובת המילקה" GridPane.columnIndex="0" GridPane.rowIndex="2" BorderPane.alignment="CENTER"/>
                                <TextField fx:id="addressField" GridPane.columnIndex="1" GridPane.rowIndex="2" BorderPane.alignment="CENTER"/>
                                <Label text="טלפון המילקה" GridPane.columnIndex="0" GridPane.rowIndex="3" BorderPane.alignment="CENTER"/>
                                <TextField fx:id="phoneField" GridPane.columnIndex="1" GridPane.rowIndex="3" BorderPane.alignment="CENTER"/>
                                <Label text="שם המנכ" GridPane.columnIndex="0" GridPane.rowIndex="4" BorderPane.alignment="CENTER"/>
                                <TextField fx:id="managerNameField" GridPane.columnIndex="1" GridPane.rowIndex="4" BorderPane.alignment="CENTER"/>
                                <Label text="טלפון המנכ" GridPane.columnIndex="0" GridPane.rowIndex="5" BorderPane.alignment="CENTER"/>
                                <TextField fx:id="managerPhoneField" GridPane.columnIndex="1" GridPane.rowIndex="5" BorderPane.alignment="CENTER"/>
                                <Label text="שם המנכ" GridPane.columnIndex="0" GridPane.rowIndex="6" BorderPane.alignment="CENTER"/>
                                <TextField fx:id="assistantNameField" GridPane.columnIndex="1" GridPane.rowIndex="6" BorderPane.alignment="CENTER"/>
                                <Label text="טלפון המנכ" GridPane.columnIndex="0" GridPane.rowIndex="7" BorderPane.alignment="CENTER"/>
                                <TextField fx:id="assistantPhoneField" GridPane.columnIndex="1" GridPane.rowIndex="7" BorderPane.alignment="CENTER"/>
                            </GridPane>
                            <Form fx:id="newDepartmentForm" BorderPane.alignment="CENTER">
                                <GridPane columns="2" BorderPane.alignment="CENTER">
                                    <Label text="שם המילקה" GridPane.columnIndex="0" GridPane.rowIndex="1" BorderPane.alignment="CENTER"/>
                                    <TextField fx:id="nameField" GridPane.columnIndex="1" GridPane.rowIndex="1" BorderPane.alignment="CENTER"/>
                                    <Label text="כתובת המילקה" GridPane.columnIndex="0" GridPane.rowIndex="2" BorderPane.alignment="CENTER"/>
                                    <TextField fx:id="addressField" GridPane.columnIndex="1" GridPane.rowIndex="2" BorderPane.alignment="CENTER"/>
                                    <Label text="טלפון המילקה" GridPane.columnIndex="0" GridPane.rowIndex="3" BorderPane.alignment="CENTER"/>
                                    <TextField fx:id="phoneField" GridPane.columnIndex="1" GridPane.rowIndex="3" BorderPane.alignment="CENTER"/>
                                    <Label text="שם המנכ" GridPane.columnIndex="0" GridPane.rowIndex="4" BorderPane.alignment="CENTER"/>
                                    <TextField fx:id="managerNameField" GridPane.columnIndex="1" GridPane.rowIndex="4" BorderPane.alignment="CENTER"/>
                                    <Label text="טלפון המנכ" GridPane.columnIndex="0" GridPane.rowIndex="5" BorderPane.alignment="CENTER"/>
                                    <TextField fx:id="managerPhoneField" GridPane.columnIndex="1" GridPane.rowIndex="5" BorderPane.alignment="CENTER"/>
                                    <Label text="שם המנכ" GridPane.columnIndex="0" GridPane.rowIndex="6" BorderPane.alignment="CENTER"/>
                                    <TextField fx:id="assistantNameField" GridPane.columnIndex="1" GridPane.rowIndex="6" BorderPane.alignment="CENTER"/>
                                    <Label text="טלפון המנכ" GridPane.columnIndex="0" GridPane.rowIndex="7" BorderPane.alignment="CENTER"/>
                                    <TextField fx:id="assistantPhoneField" GridPane.columnIndex="1" GridPane.rowIndex="7" BorderPane.alignment="CENTER"/>
                                </GridPane>
                            </Form>
                        </Form>
                    </center>
                    <bottom>
                        <Form fx:id="cancelForm" BorderPane.alignment="CENTER">
                            <GridPane columns="2" BorderPane.alignment="CENTER">
                                <Label text="האם אתה בטוח?" GridPane.columnIndex="0" GridPane.rowIndex="1" BorderPane.alignment="CENTER"/>
                                <Button text="ביטול" GridPane.columnIndex="1" GridPane.rowIndex="1" BorderPane.alignment="CENTER"/>
                            </GridPane>
                        </Form>
                    </bottom>
                </BorderPane>
            </children>
        </AnchorPane>
    </children>
</AnchorPane>
```

```

                </effect>
            </Label>
        </top>
        <bottom>
            <ButtonBar prefHeight="40.0" prefWidth="200.0"
BorderPane.alignment="CENTER">
                <buttons>
                    <ButtonBar prefHeight="40.0" prefWidth="200.0">
                        <buttons>
                            <Button mnemonicParsing="false"
onAction="#saveDepartmentClicked" text="גִּנְוָשׁ" textFill="#279a51">
                                <font>
                                    <Font size="14.0" />
                                </font>
                            </Button>
                            <Button mnemonicParsing="false"
onAction="#backToManageDepartmentClick" text="בָּרְכַּת" textFill="#f50707">
                                <font>
                                    <Font size="14.0" />
                                </font>
                            </Button>
                        </buttons>
                    <effect>
                        <MotionBlur angle="288.01" radius="0.0">
                            <input>
                                <InnerShadow />
                            </input>
                        </MotionBlur>
                    </effect>
                </ButtonBar>
            </buttons>
        </ButtonBar>
    </bottom>
    <left>
        <ImageView fitHeight="329.0" fitWidth="169.0"
pickOnBounds="true" preserveRatio="true" BorderPane.alignment="CENTER">
            <image>
                <Image
url="@../../../../Images/backgroundImage.jpg" />
            </image>
        </ImageView>
    </left>
    <center>
        <GridPane BorderPane.alignment="CENTER">
            <columnConstraints>
                <ColumnConstraints hgrow="SOMETIMES" maxWidth="242.0"
minWidth="10.0" prefWidth="242.0" />
                <ColumnConstraints hgrow="SOMETIMES" maxWidth="178.0"
minWidth="10.0" prefWidth="124.0" />
            </columnConstraints>
            <rowConstraints>
                <RowConstraints minHeight="10.0" prefHeight="30.0"
vgrow="SOMETIMES" />
                <RowConstraints minHeight="10.0" prefHeight="30.0"
vgrow="SOMETIMES" />
            </rowConstraints>
            <children>
                <Label contentDisplay="CENTER" text="מִזְלְקָה :"
GridPane.columnIndex="1" GridPane.halignment="CENTER" />
                <Label contentDisplay="CENTER" text="מִזְלָקָה תֵּיאֹרֶה / "
GridPane.columnIndex="1" GridPane.halignment="CENTER"
            <Label contentDisplay="CENTER" text="הָעֲרָה :"
GridPane.columnIndex="1" GridPane.halignment="CENTER" />
            </children>
        </GridPane>
    </center>

```

```

GridPane.rowIndex="1" />
    <TextField fx:id="tf_departmentName">
        <GridPane.margin>
            <Insets left="20.0" right="20.0" />
        </GridPane.margin>
    </TextField>
    <TextField fx:id="tf_description"
GridPane.rowIndex="1">
    <GridPane.margin>
        <Insets left="20.0" right="20.0" />
    </GridPane.margin>
    </TextField>
</children>
</GridPane>
</center>
</BorderPane>
</children>
</AnchorPane>
</children>
</AnchorPane>

```



editDepartment_view.fxml

```

<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.geometry.Insets?>
<?import javafx.scene.control.Button?>
<?import javafx.scene.control.ChoiceBox?>
<?import javafx.scene.control.Label?>
<?import javafx.scene.control.TextField?>
<?import javafx.scene.effect.InnerShadow?>
<?import javafx.scene.effect.Light.Distant?>
<?import javafx.scene.effect.Lighting?>
<?import javafx.scene.layout.AnchorPane?>
<?import javafx.scene.layout.ColumnConstraints?>
<?import javafx.scene.layout.GridPane?>
<?import javafx.scene.layout.RowConstraints?>
<?import javafx.scene.paint.Color?>
<?import javafx.scene.text.Font?>

<AnchorPane prefHeight="459.0" prefWidth="655.0" style="-fx-background-
color: grey;" xmlns="http://javafx.com/javafx/19"
xmlns:fx="http://javafx.com/fxml/1"
fx:controller="com.example.hrpulse.Controllers.DepartmentController.EditDep-
artmentController">
    <children>
        <GridPane layoutX="274.0" layoutY="217.0"
AnchorPane.bottomAnchor="10.0" AnchorPane.leftAnchor="10.0"
AnchorPane.rightAnchor="10.0" AnchorPane.topAnchor="10.0">
            <columnConstraints>
                <ColumnConstraints hgrow="SOMETIMES" minWidth="10.0"
prefWidth="100.0" />
            </columnConstraints>
            <rowConstraints>
                <RowConstraints minHeight="10.0" prefHeight="30.0"
vgrow="SOMETIMES" />
            </rowConstraints>
            <children>
                <GridPane>
```

```

        <columnConstraints>
            <ColumnConstraints hgrow="SOMETIMES" minWidth="10.0"
prefWidth="100.0" />
        </columnConstraints>
        <rowConstraints>
            <RowConstraints minHeight="10.0" prefHeight="10.0"
valignment="CENTER" vgrow="SOMETIMES" />
            <RowConstraints minHeight="10.0" prefHeight="30.0"
vgrow="SOMETIMES" />
            <RowConstraints minHeight="10.0" prefHeight="30.0"
vgrow="SOMETIMES" />
        </rowConstraints>
        <GridPane.margin>
            <Insets />
        </GridPane.margin>
        <children>
            <Label text="מזהה צוות נציגות " GridPane.halignment="CENTER">
                <font>
                    <Font size="14.0" />
                </font>
            </Label>
            <GridPane GridPane.rowIndex="1">
                <columnConstraints>
                    <ColumnConstraints hgrow="SOMETIMES" minWidth="10.0"
prefWidth="100.0" />
                    <ColumnConstraints hgrow="SOMETIMES" maxWidth="150.0"
minWidth="100.0" prefWidth="100.0" />
                </columnConstraints>
                <rowConstraints>
                    <RowConstraints minHeight="10.0" prefHeight="30.0"
vgrow="SOMETIMES" />
                    <RowConstraints minHeight="10.0" prefHeight="30.0"
vgrow="SOMETIMES" />
                    <RowConstraints minHeight="10.0" prefHeight="30.0"
vgrow="SOMETIMES" />
                </rowConstraints>
                <children>
                    <Label text="שם בחר מזהה :" GridPane.columnIndex="1"
GridPane.halignment="CENTER" />
                    <ChoiceBox fx:id="cb_departmentShow"
maxWidth="200.0" minWidth="150.0" prefWidth="150.0"
GridPane.halignment="RIGHT" />
                    <Label fx:id="label_departmentName" text="שם מזהה :"
GridPane.columnIndex="1" GridPane.rowIndex="1" />
                    <Label fx:id="label_departmentDescription"
text="תיאור מזהה :" GridPane.columnIndex="1"
GridPane.rowIndex="2" />
                    <TextField fx:id="tf_department" maxWidth="250.0"
minWidth="150.0" GridPane.halignment="RIGHT"
GridPane.rowIndex="1" />
                    <TextField fx:id="tf_departmentDescription"
maxWidth="250.0" minWidth="150.0" GridPane.halignment="RIGHT"
GridPane.rowIndex="2" />
                </children>
            </GridPane>
            <GridPane GridPane.rowIndex="2">
                <columnConstraints>
                    <ColumnConstraints hgrow="SOMETIMES"
maxWidth="100.0" minWidth="10.0" prefWidth="100.0" />
                    <ColumnConstraints hgrow="SOMETIMES" minWidth="10.0"
prefWidth="100.0" />
                    <ColumnConstraints hgrow="SOMETIMES" minWidth="10.0"

```

```

prefWidth="100.0" />
        </columnConstraints>
        <rowConstraints>
            <RowConstraints minHeight="10.0" prefHeight="30.0"
vgrow="SOMETIMES" />
            </rowConstraints>
            <children>
                <Button maxHeight="30.0" maxWidth="150.0"
minHeight="30.0" minWidth="100.0" mnemonicParsing="false"
onAction="#backButtonClicked" text="戻る" GridPane.columnIndex="2"
GridPane.halignment="CENTER">
                    <effect>
                        <Lighting>
                            <bumpInput>
                                <InnerShadow />
                            </bumpInput>
                            <light>
                                <Light.Distant>
                                    <color>
                                        <Color red="0.5550604462623596"
green="0.8105263113975525" blue="0.3962573110572815" />
                                    </color>
                                </Light.Distant>
                            </light>
                        </Lighting>
                    </effect>
                </Button>
                <Button fx:id="btn_editButton" maxHeight="60.0"
maxWidth="60.0" minHeight="60.0" minWidth="60.0" mnemonicParsing="false"
onAction="#editButtonClicked" text="編集" GridPane.columnIndex="1"
GridPane.halignment="LEFT">
                    <effect>
                        <InnerShadow>
                            <color>
                                <Color red="0.03118908405303955"
green="0.5263158082962036" blue="0.16322287917137146" />
                            </color>
                        </InnerShadow>
                    </effect>
                </Button>
                <Button fx:id="btn_removeButton" maxHeight="60.0"
maxWidth="60.0" minHeight="60.0" minWidth="60.0" mnemonicParsing="false"
onAction="#removeButtonClicked" text="削除" GridPane.halignment="CENTER">
                    <effect>
                        <InnerShadow>
                            <color>
                                <Color red="0.7894737124443054"
green="0.09356725215911865" blue="0.09356725215911865" />
                            </color>
                        </InnerShadow>
                    </effect>
                </Button>
            </children>
        </GridPane>
        <children>
            </GridPane>
        </children>
    </GridPane>
</children>
</AnchorPane>

```



manageDepartment_view.fxml

```
<?xml version="1.0" encoding="UTF-8"?>
<?import javafx.scene.control.Button?>
<?import javafx.scene.control.Label?>
<?import javafx.scene.effect.InnerShadow?>
<?import javafx.scene.image.Image?>
<?import javafx.scene.image.ImageView?>
<?import javafx.scene.layout.AnchorPane?>
<?import javafx.scene.layout.BorderPane?>
<?import javafx.scene.text.Font?>

<AnchorPane prefHeight="696.0" prefWidth="1071.0"
xmlns="http://javafx.com/javafx/20.0.1" xmlns:fx="http://javafx.com/fxml/1"
fx:controller="com.example.hrpulse.Controllers.DepartmentController.Managed
eprtementController">
    <children>
        <BorderPane layoutX="71.0" layoutY="63.0" prefHeight="593.0"
prefWidth="954.0">
            <top>
                <Label text="מזהם מילוקי" textFill="#5545eb"
BorderPane.alignment="CENTER">
                    <font>
                        <Font name="System Bold Italic" size="24.0" />
                    </font>
                </Label>
            </top>
            <bottom>
                <Button mnemonicParsing="false" onAction="#BackToMainClicked"
prefHeight="33.0" prefWidth="248.0" text="הזרה"
BorderPane.alignment="CENTER">
                    <font>
                        <Font name="Calisto MT Bold Italic" size="14.0" />
                    </font>
                </Button>
            </bottom>
            <right>
                <Button mnemonicParsing="false" prefHeight="569.0"
prefWidth="52.0" BorderPane.alignment="CENTER" />
            </right>
            <center>
                <AnchorPane prefHeight="535.0" prefWidth="866.0"
BorderPane.alignment="CENTER">
                    <children>
                        <Button layoutX="109.0" layoutY="118.0"
mnemonicParsing="false" onAction="#createDepartmentClicked"
prefHeight="50.0" prefWidth="400.0" style="-fx-background-color: #2a6a6f;" text="הוסף נייחות"
textFill="#e3ebef" AnchorPane.bottomAnchor="357.0"
AnchorPane.rightAnchor="124.0" >
                            <font>
                                <Font name="System Bold Italic" size="14.0" />
                            </font>
                        </Button>
                        <Button alignment="CENTER" layoutX="107.0"
layoutY="216.0" mnemonicParsing="false"
onAction="#navigateToEditDepartment" prefHeight="50.0" prefWidth="400.0"
style="-fx-background-color: #2a6a6f;" text="עדכון הטרה / מילוקיות"
textFill="#e3ebef" AnchorPane.bottomAnchor="357.0"
AnchorPane.rightAnchor="124.0" >
                            <font>
                                <Font name="System Bold Italic" size="14.0" />
                            </font>
                        </Button>
                    </children>
                </AnchorPane>
            </center>
        </BorderPane>
    </children>

```

```

textFill="#fffffe" AnchorPane.bottomAnchor="259.0"
AnchorPane.rightAnchor="126.0">
    <font>
        <Font name="System Bold Italic" size="14.0" />
    </font>
</Button>
<Button layoutX="107.0" layoutY="329.0"
mnemonicParsing="false" onAction="#navigateToDepartmentDetails"
prefHeight="50.0" prefWidth="400.0" style="-fx-background-color: #2a6a6f;" text="המחלקות ברכי"
textFill="#fcf8f8">
    <font>
        <Font name="System Bold Italic" size="14.0" />
    </font>
</Button>
</children>
<effect>
    <InnerShadow />
</effect>
</AnchorPane>
</center>
<left>
    <ImageView fitHeight="96.0" fitWidth="362.0"
pickOnBounds="true" preserveRatio="true" BorderPane.alignment="CENTER">
        <image>
            <Image url="@../../../../Images/HR_logo.jpg" />
        </image>
    </ImageView>
</left>
</BorderPane>
</children>
</AnchorPane>

```



reportDepartment_view.fxml

```

<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.geometry.Insets?>
<?import javafx.scene.control.Button?>
<?import javafx.scene.control.Label?>
<?import javafx.scene.control.ListView?>
<?import javafx.scene.effect.InnerShadow?>
<?import javafx.scene.layout.BorderPane?>
<?import javafx.scene.paint.Color?>
<?import javafx.scene.text.Font?>
<?import com.example.hrpulse.DepartmentCellFactory?>

<BorderPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-
Infinity" minWidth="-Infinity" prefHeight="400.0" prefWidth="600.0"
xmlns="http://javafx.com/javafx/19" xmlns:fx="http://javafx.com/fxml/1"
fx:controller="com.example.hrpulse.Controllers.DepartmentController.ReportD
epartmentController">
    <top>
        <Label alignment="CENTER" text="המחלקות ברכי" BorderPane.alignment="CENTER">
            <font>
                <Font name="Comic Sans MS" size="16.0" />

```

```

</font>
<effect>
    <InnerShadow>
        <color>
            <Color red="0.29674848914146423"
green="0.49000000953674316" blue="0.08820000290870667" />
        </color>
    </InnerShadow>
</effect>
<BorderPane.margin>
    <Insets top="15.0" />
</BorderPane.margin>
</Label>
</top>
<bottom>
    <Button mnemonicParsing="false" onAction="#backToManageDepartment"
text="戻る" BorderPane.alignment="CENTER">
        <font>
            <Font name="Comic Sans MS" size="14.0" />
        </font>
        <effect>
            <InnerShadow>
                <color>
                    <Color red="0.216000056028366"
green="0.36000001430511475" blue="0.2311675101518631" />
                </color>
            </InnerShadow>
        </effect>
        <BorderPane.margin>
            <Insets bottom="15.0" />
        </BorderPane.margin>
    </Button>
</bottom>
<center>
    <ListView fx:id="lv_departments" prefHeight="200.0" prefWidth="200.0"
BorderPane.alignment="CENTER">
        <cellFactory>
            <DepartmentCellFactory />
        </cellFactory>
    </ListView>
</center>
</BorderPane>

```




createNewEmployee_view.fxml createNewEmployee.css

```

<?xml version="1.0" encoding="UTF-8"?>

<?import java.lang.String?>
<?import javafx.collections.FXCollections?>
<?import javafx.geometry.Insets?>
<?import javafx.scene.control.Button?>
<?import javafx.scene.control.ButtonBar?>
<?import javafx.scene.control.CheckBox?>
<?import javafx.scene.control.ChoiceBox?>
<?import javafx.scene.control.DatePicker?>
<?import javafx.scene.control.Label?>
<?import javafx.scene.control.TextField?>
<?import javafx.scene.layout.BorderPane?>
<?import javafx.scene.layout.ColumnConstraints?>
<?import javafx.scene.layout.GridPane?>
<?import javafx.scene.layout.RowConstraints?>
```

```

<?import javafx.scene.text.Font?>

<BorderPane prefHeight="734.0" prefWidth="975.0" styleClass="root"
stylesheets="@createNewEmployee.css" xmlns="http://javafx.com/javafx/19"
xmlns:fx="http://javafx.com/fxml/1"
fx:controller="com.example.hrpulse.Controllers.EmployeeController.CreateEmployeePageController">
    <center>
        <GridPane alignment="CENTER" prefHeight="521.0" prefWidth="832.0"
BorderPane.alignment="CENTER">
            <columnConstraints>
                <ColumnConstraints halignment="RIGHT" hgrow="SOMETIMES"
maxWidth="652.0" minWidth="10.0" prefWidth="652.0" />
                <ColumnConstraints halignment="CENTER" hgrow="SOMETIMES"
maxWidth="494.0" minWidth="10.0" prefWidth="347.0" />
            </columnConstraints>
            <rowConstraints>
                <RowConstraints minHeight="10.0" prefHeight="30.0" />
                <RowConstraints minHeight="10.0" prefHeight="30.0"
vgrow="SOMETIMES" />
            </rowConstraints>
            <children>
                <Label text="מַשְׁנָה :" GridPane.columnIndex="1"
GridPane.rowIndex="1" />
                <Label text="מַשְׁנָה :" GridPane.columnIndex="1"
GridPane.rowIndex="2" />
                <Label text="נִ.ת :" GridPane.columnIndex="1"
GridPane.rowIndex="3" />
                <Label text="רִתְקָוּת :" GridPane.columnIndex="1"

```

```

GridPane.rowIndex="4" />
    <Label text="טליל מס' :" GridPane.columnIndex="1"
GridPane.rowIndex="6" />
    <Label text="מגן :" GridPane.columnIndex="1"
GridPane.rowIndex="5" />
    <Label text="תפקיד :" GridPane.columnIndex="1"
GridPane.rowIndex="7" />
    <Label text="מחלקה :" GridPane.columnIndex="1"
GridPane.rowIndex="8" />
    <Label text="כתובת :" GridPane.columnIndex="1"
GridPane.rowIndex="11" />
    <Label text="טלפון :" GridPane.columnIndex="1"
GridPane.rowIndex="12" />
    <Label text="בנק מס' :" GridPane.columnIndex="1"
GridPane.rowIndex="14" />
    <Label text="סניף מס' :" GridPane.columnIndex="1"
GridPane.rowIndex="15" />
    <TextField fx:id="tf_sneefBankCode" maxWidth="200.0"
prefHeight="25.0" prefWidth="627.0" GridPane.rowIndex="15" />
    <Label text="טליל מס' :" GridPane.columnIndex="1"
GridPane.rowIndex="16" />
    <TextField fx:id="tf_accountNumber" maxWidth="200.0"
prefHeight="25.0" prefWidth="627.0" GridPane.rowIndex="16" />
    <GridPane GridPane.rowIndex="1">
        <columnConstraints>
            <ColumnConstraints hgrow="SOMETIMES" maxWidth="450.0"
minWidth="10.0" prefWidth="450.0" />
            <ColumnConstraints hgrow="SOMETIMES" maxWidth="320.0"
minWidth="10.0" prefWidth="202.0" />
        </columnConstraints>
        <rowConstraints>
            <RowConstraints minHeight="10.0" prefHeight="30.0"
vgrow="SOMETIMES" />
        </rowConstraints>
        <children>
            <TextField fx:id="tf(firstName" alignment="CENTER_RIGHT"
maxWidth="200.0" prefHeight="25.0" prefWidth="627.0"
GridPane.columnIndex="1" GridPane.halignment="RIGHT" />
            <Label text="*" textFill="#f2290a"
GridPane.halignment="RIGHT">
                <font>
                    <Font size="18.0" />
                </font>
            </Label>
        </children>
    </GridPane>
    <GridPane GridPane.rowIndex="2">
        <columnConstraints>
            <ColumnConstraints hgrow="SOMETIMES" maxWidth="451.0"
minWidth="10.0" prefWidth="451.0" />
            <ColumnConstraints hgrow="SOMETIMES" maxWidth="321.0"
minWidth="10.0" prefWidth="201.0" />
        </columnConstraints>
        <rowConstraints>
            <RowConstraints minHeight="10.0" prefHeight="30.0"
vgrow="SOMETIMES" />
        </rowConstraints>
        <children>
            <TextField fx:id="tf(lastName" maxWidth="200.0"
prefHeight="25.0" prefWidth="151.0" GridPane.columnIndex="1" />
            <Label text="*" textFill="#f2290a"

```

```
GridPane.alignment="RIGHT">
    <font>
        <Font size="18.0" />
    </font>
</Label>
</children>
</GridPane>
<GridPane GridPane.rowIndex="3">
    <columnConstraints>
        <ColumnConstraints hgrow="SOMETIMES" maxWidth="452.0"
minWidth="10.0" prefWidth="452.0" />
        <ColumnConstraints hgrow="SOMETIMES" maxWidth="321.0"
minWidth="10.0" prefWidth="200.0" />
    </columnConstraints>
    <rowConstraints>
        <RowConstraints minHeight="10.0" prefHeight="30.0"
vgrow="SOMETIMES" />
    </rowConstraints>
    <children>
        <TextField fx:id="tf_employeeID" maxWidth="200.0"
prefHeight="25.0" prefWidth="627.0" GridPane.columnIndex="1"
GridPane.alignment="RIGHT" />
        <Label text="*" textFill="#f2290a" />
    </children>
</GridPane>
<GridPane GridPane.rowIndex="4">
    <columnConstraints>
        <ColumnConstraints hgrow="SOMETIMES" maxWidth="452.0"
minWidth="10.0" prefWidth="452.0" />
        <ColumnConstraints hgrow="SOMETIMES" maxWidth="321.0"
minWidth="10.0" prefWidth="200.0" />
    </columnConstraints>
    <rowConstraints>
        <RowConstraints minHeight="10.0" prefHeight="30.0"
vgrow="SOMETIMES" />
    </rowConstraints>
    <children>
        <TextField fx:id="tf_email" maxWidth="200.0"
prefHeight="25.0" prefWidth="627.0" GridPane.columnIndex="1"
GridPane.alignment="RIGHT" />
        <Label text="*" textFill="#f2290a" />
    </children>
</GridPane>
<GridPane GridPane.rowIndex="6">
    <columnConstraints>
        <ColumnConstraints hgrow="SOMETIMES" maxWidth="448.0"
minWidth="10.0" prefWidth="448.0" />
        <ColumnConstraints hgrow="SOMETIMES" maxWidth="319.0"
minWidth="10.0" prefWidth="204.0" />
    </columnConstraints>
    <rowConstraints>
```

```

        <RowConstraints minHeight="10.0" prefHeight="30.0"
vgrow="SOMETIMES" />
    </rowConstraints>
    <children>
        <Label text="*" textFill="#f2290a"
GridPane.halignment="RIGHT">
            <font>
                <Font size="18.0" />
            </font>
        </Label>
        <TextField fx:id="tf_phoneNumber" maxWidth="200.0"
prefHeight="25.0" prefWidth="627.0" GridPane.columnIndex="1" />
    </children>
</GridPane>
<ChoiceBox fx:id="cb_gender" minWidth="80.0" prefHeight="25.0"
prefWidth="52.0" GridPane.rowIndex="5">
    <items>
        <FXCollections fx:factory="observableArrayList">
            <String fx:value="זכר" />
            <String fx:value="נקבה" />
        </FXCollections>
    </items>
</ChoiceBox>
<ChoiceBox fx:id="cb_department" minWidth="80.0"
prefHeight="25.0" prefWidth="52.0" GridPane.rowIndex="8" />
    <Label text="לידת חארץ" GridPane.columnIndex="1"
GridPane.rowIndex="9" />
    <GridPane GridPane.rowIndex="11">
        <columnConstraints>
            <ColumnConstraints hgrow="SOMETIMES" maxWidth="467.8"
minWidth="10.0" prefWidth="462.2000244140625" />
            <ColumnConstraints hgrow="SOMETIMES" maxWidth="200.0"
minWidth="10.0" prefWidth="101.59997558593744" />
            <ColumnConstraints hgrow="SOMETIMES" maxWidth="200.0"
minWidth="10.0" prefWidth="45.39998779296877" />
        </columnConstraints>
        <rowConstraints>
            <RowConstraints minHeight="10.0" prefHeight="30.0"
vgrow="SOMETIMES" />
        </rowConstraints>
        <children>
            <TextField fx:id="tf_salaryPerHour" maxWidth="100.0"
prefHeight="25.0" prefWidth="627.0" GridPane.halignment="RIGHT" />
            <CheckBox fx:id="cb_isHourly" mnemonicParsing="false"
GridPane.columnIndex="2" GridPane.halignment="RIGHT" />
            <Label text="₪ שעה :" GridPane.columnIndex="1"
GridPane.halignment="LEFT" />
        </children>
    </GridPane>
    <TextField fx:id="tf_bankNumber" maxWidth="200.0"
prefHeight="25.0" prefWidth="627.0" GridPane.rowIndex="14" />
    <Label alignment="CENTER" contentDisplay="CENTER"
prefHeight="41.0" prefWidth="180.0" text="תובד חכמת שדיי">
        <font>
            <Font name="System Bold Italic" size="18.0" />
        </font>
    </Label>
    <GridPane GridPane.rowIndex="17">
        <columnConstraints>
            <ColumnConstraints halignment="CENTER" hgrow="SOMETIMES"
minWidth="10.0" prefWidth="100.0" />

```

```

        <ColumnConstraints halignment="CENTER" hgrow="SOMETIMES"
minWidth="10.0" prefWidth="100.0" />
    </columnConstraints>
    <rowConstraints>
        <RowConstraints minHeight="10.0" prefHeight="30.0"
vgrow="SOMETIMES" />
    </rowConstraints>
</GridPane>
<Label text="תְּלִינָה יְמִינָה :" GridPane.columnIndex="1"
GridPane.halignment="CENTER" GridPane.rowIndex="13" />
<TextField fx:id="tf_salaryToTravel" maxWidth="200.0"
prefHeight="25.0" prefWidth="627.0" GridPane.halignment="RIGHT"
GridPane.rowIndex="13" />
<DatePicker fx:id="dp_dateOfBirth" maxWidth="200.0"
prefHeight="25.0" prefWidth="627.0" GridPane.rowIndex="9" />
<GridPane GridPane.rowIndex="7">
    <columnConstraints>
        <ColumnConstraints hgrow="SOMETIMES"
maxWidth="525.6000122070312" minWidth="10.0" prefWidth="525.6000122070312"
/>
        <ColumnConstraints hgrow="SOMETIMES"
maxWidth="319.60003662109375" minWidth="80.0"
prefWidth="127.19998779296873" />
    </columnConstraints>
    <rowConstraints>
        <RowConstraints minHeight="10.0" prefHeight="30.0"
vgrow="SOMETIMES" />
    </rowConstraints>
    <children>
        <ChoiceBox fx:id="cb_role" minWidth="80.0"
prefHeight="25.0" prefWidth="52.0" GridPane.columnIndex="1"
GridPane.halignment="RIGHT">
            <items>
                <FXCollections fx:factory="observableArrayList">
                    <String fx:value="secretary" />
                    <String fx:value="headOfDepartment" />
                    <String fx:value="employee" />
                </FXCollections>
            </items>
        </ChoiceBox>
        <TextField fx:id="tf_password" alignment="CENTER_RIGHT"
disable="true" maxWidth="160.0" promptText="סיסמה בחר נא"
GridPane.halignment="RIGHT" />
    </children>
</GridPane>
<GridPane GridPane.rowIndex="12">
    <columnConstraints>
        <ColumnConstraints hgrow="SOMETIMES"
maxWidth="590.399951171875" minWidth="10.0" prefWidth="534.3999755859375"
/>
        <ColumnConstraints hgrow="SOMETIMES" maxWidth="200.0"
minWidth="10.0" prefWidth="534.3999755859375" />
        <ColumnConstraints hgrow="SOMETIMES" maxWidth="200.0"
minWidth="10.0" prefWidth="117.60002441406255" />
    </columnConstraints>
    <rowConstraints>
        <RowConstraints minHeight="10.0" prefHeight="30.0"
vgrow="SOMETIMES" />
    </rowConstraints>
    <children>
        <CheckBox fx:id="cb_isPerMoth" mnemonicParsing="false"

```

```

GridPane.columnIndex="2" GridPane.halignment="RIGHT" />
    <TextField fx:id="tf_perMonth" maxWidth="100.0" />
GridPane.halignment="RIGHT" />
    <Label text="זינור שכר :" GridPane.columnIndex="1" />
</children>
</GridPane>
<Label text="עובדת תחילת תאריך :" GridPane.columnIndex="1" />
GridPane.rowIndex="10" />
    <DatePicker fx:id="dp(startDate" maxWidth="200.0" />
prefHeight="25.0" prefWidth="627.0" GridPane.rowIndex="10" />
    </children>
</GridPane>
</center>
<bottom>
    <ButtonBar prefHeight="40.0" prefWidth="200.0" />
BorderPane.alignment="CENTER">
    <buttons>
        <Button fx:id="saveButton" maxHeight="40.0" maxWidth="150.0" minHeight="40.0" minWidth="150.0" mnemonicParsing="false" onAction="#saveButtonClicked" prefHeight="35.0" prefWidth="122.0" text="מירה" >
            <font>
                <Font name="System Bold Italic" size="16.0" />
            </font>
        </Button>
        <Button fx:id="backButton" maxHeight="40.0" maxWidth="150.0" minHeight="40.0" minWidth="150.0" mnemonicParsing="false" onAction="#goToMain" text="זרזע" >
            <font>
                <Font name="System Bold Italic" size="16.0" />
            </font>
        </Button>
    </buttons>
    <BorderPane.margin>
        <Insets />
    </BorderPane.margin>
    <padding>
        <Insets bottom="20.0" right="50.0" />
    </padding>
</ButtonBar>
</bottom>
</BorderPane>

.root {
    -fx-background-color: #f0f0f0; /* Set the background color */
    -fx-font-family: "Arial"; /* Change the default font-family */
    -fx-font-size: 14px; /* Change the default font size */
}

/* Style for labels */
.label {
    -fx-font-size: 16px; /* Increase label font size */
    -fx-text-fill: #000000; /* Set label text color to black */
}

/* Style for text fields */
.text-field {
    -fx-font-size: 14px;

```

```

}

/* Style for choice boxes */
.choice-box {
    -fx-font-size: 14px;
}

/* Style for buttons */
.button {
    -fx-font-size: 16px;
    -fx-text-fill: #ffffff; /* Set button text color to white */
    -fx-background-color: #4CAF50; /* Default button background color */
}

/* Button hover effect */
.button:hover {
    -fx-background-color: #45a049; /* Button background color on hover */
    -fx-effect: innershadow(three-pass-box, #d8d8d8, 10, 0, 0, 0);
}

/* Style for date pickers */
.date-picker {
    -fx-font-size: 14px;
}

/* Style for checkboxes */
.checkbox {
    -fx-font-size: 14px;
}

```



editEmployeeShift_view.fxml

```

<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.control.Button?>
<?import javafx.scene.control.Label?>
<?import javafx.scene.control.TableColumn?>
<?import javafx.scene.control.TableView?>
<?import javafx.scene.control.TextField?>
<?import javafx.scene.effect.InnerShadow?>
<?import javafx.scene.layout.AnchorPane?>
<?import javafx.scene.layout.ColumnConstraints?>
<?import javafx.scene.layout.GridPane?>
<?import javafx.scene.layout.HBox?>
<?import javafx.scene.layout.RowConstraints?>
<?import javafx.scene.layout.VBox?>
<?import javafx.scene.text.Font?>

<AnchorPane prefHeight="600.0" prefWidth="1250.0"
xmlns="http://javafx.com/javafx/21" xmlns:fx="http://javafx.com/fxml/1"
fx:controller="com.example.hrpulse.Controllers.EmployeeController.EditEmployeeShiftController">
    <children>
        <VBox alignment="CENTER" layoutX="400" layoutY="14.0" spacing="20">
            <AnchorPane.leftAnchor="300.0" AnchorPane.rightAnchor="300.0"
            AnchorPane.topAnchor="14.0">
                <Label prefHeight="40.0" text="הוֹצֵא שְׁמוֹן דָּבָר">

```

```

        <font>
            <Font name="System Bold Italic" size="24.0" />
        </font>
    </Label>
    <HBox alignment="CENTER_RIGHT" spacing="20" />
</VBox>

<TableView fx:id="tableViewCSVData" layoutX="51.0" layoutY="100.0"
minWidth="400" prefHeight="260.0" prefWidth="1085.0"
AnchorPane.bottomAnchor="200.0" AnchorPane.leftAnchor="51.0"
AnchorPane.topAnchor="140.0">
    <columns>
        <TableColumn fx:id="tc_totalWorkHrs" maxWidth="305.0"
minWidth="100.0" prefWidth="192.0" text="תֹּאכְלִימָה" />
        <TableColumn fx:id="tc_breakTime" maxWidth="367.0"
minWidth="100.0" prefWidth="183.0" text="עֲמָקָם הַקְרָבָה" />
        <TableColumn fx:id="tc_exitHour" maxWidth="427.0"
minWidth="100.0" prefWidth="163.0" text="עַזְבֵּת הַשְׁעָרָה" />
        <TableColumn fx:id="tc_enterHour" maxWidth="371.0"
minWidth="86.0" prefWidth="156.0" text="הַשְׁעָרָה כְּבָשָׂר" />
        <TableColumn fx:id="tc_date" maxWidth="351.0"
minWidth="127.0" prefWidth="141.0" text="לְמַעַתְּמָה" />
        <TableColumn fx:id="tc_employeeId" maxWidth="362.0"
minWidth="73.99993896484375" prefWidth="151.0" text="אַתְּ" />
    </columns>
</TableView>

<GridPane layoutX="50.0" layoutY="400.0"
AnchorPane.bottomAnchor="60.0" AnchorPane.leftAnchor="50.0"
AnchorPane.rightAnchor="50.0">
    <HBox alignment="CENTER_RIGHT" prefWidth="265.0" spacing="20"
/>
    <columnConstraints>
        <ColumnConstraints />
    </columnConstraints>
    <rowConstraints>
        <RowConstraints />
    </rowConstraints>
</GridPane>

<GridPane alignment="CENTER_RIGHT" layoutX="50.0" layoutY="480.0"
AnchorPane.bottomAnchor="160.0" AnchorPane.leftAnchor="50.0"
AnchorPane.rightAnchor="50.0">
    <Label fx:id="tf_countDays" maxWidth="60.0" minWidth="60.0">
        <effect>
            <InnerShadow />
        </effect>
    </Label>
    <Label fx:id="tf_countHours" maxWidth="60.0" minWidth="60.0"
GridPane.columnIndex="2" />
    <columnConstraints>
        <ColumnConstraints />
        <ColumnConstraints />
        <ColumnConstraints />
        <ColumnConstraints />
    </columnConstraints>
    <rowConstraints>
        <RowConstraints />
    </rowConstraints>
</GridPane>

```

```

        <Button layoutX="1018.0" layoutY="400.0" onAction="#uploadCsvFile"
text="העלאת נתונים קובץ CSV" />
        <Button layoutX="983.0" layoutY="508.0" mnemonicParsing="false"
onAction="#mainPageButtonClicked" prefHeight="40.0" prefWidth="150.0"
text="דף ראשי" AnchorPane.leftAnchor="983.0">
            <font>
                <Font name="System Bold Italic" size="18.0" />
            </font>
        </Button>

        <!-- Add the button for loading last saved data from the database --
->
        <Button fx:id="loadLastSavedDataButton" layoutX="833.0"
layoutY="400.0" onAction="#loadLastSavedDataIntoTableview" text="חישור נתונים שמשרדו"
/>

        <Button fx:id="saveButton" layoutX="50.0" layoutY="508.0"
mnemonicParsing="false" onAction="#saveButtonClicked" prefHeight="40.0"
prefWidth="150.0" text="שמירה">
            <font>
                <Font name="System Bold Italic" size="18.0" />
            </font>
        </Button>
        <Button layoutX="685.0" layoutY="400.0"
onAction="#addRowButtonClicked" text="הוספה חדשה שורה" />
        <Label layoutX="1088.0" layoutY="110.0" text="ט.ז שובץ:" />
        <TextField fx:id="tf_employeeID" layoutX="927.0" layoutY="106.0"
prefWidth="150.0" promptText="ט.ז חדש שובץ" />
        <Button layoutX="879.0" layoutY="106.0" mnemonicParsing="false"
onAction="#searchButtonClicked" text="חיפוש" />
    </children>
</AnchorPane>

```

 editEmployee_view.fxml

 editEmployee.css

```

<?xml version="1.0" encoding="UTF-8"?>

<?import java.lang.String?>
<?import javafx.collections.FXCollections?>
<?import javafx.geometry.Insets?>
<?import javafx.scene.control.Button?>
<?import javafx.scene.control.ButtonBar?>
<?import javafx.scene.control.CheckBox?>
<?import javafx.scene.control.ChoiceBox?>
<?import javafx.scene.control.Label?>
<?import javafx.scene.control.TextField?>
<?import javafx.scene.effect.DropShadow?>
<?import javafx.scene.effect.InnerShadow?>
<?import javafx.scene.layout.BorderPane?>
<?import javafx.scene.layout.ColumnConstraints?>
<?import javafx.scene.layout.GridPane?>
<?import javafx.scene.layout.RowConstraints?>
<?import javafx.scene.paint.Color?>

```

```

<?import javafx.scene.text.Font?>

<BorderPane prefHeight="734.0" prefWidth="975.0" styleClass="root"
stylesheets="@../CreateNewEmployeeView/createNewEmployee.css"
xmlns="http://javafx.com/javafx/19" xmlns:fx="http://javafx.com/fxml/1"
fx:controller="com.example.hrpulse.Controllers.EmployeeController.EditEmployeeController">
    <center>
        <GridPane fx:id="gp_editEmployee" alignment="CENTER"
prefHeight="521.0" prefWidth="832.0" BorderPane.alignment="CENTER">
            <columnConstraints>
                <ColumnConstraints halignment="RIGHT" hgrow="SOMETIMES"
maxWidth="652.0" minWidth="10.0" prefWidth="652.0" />
                <ColumnConstraints halignment="CENTER" hgrow="SOMETIMES"
maxWidth="494.0" minWidth="10.0" prefWidth="347.0" />
            </columnConstraints>
            <rowConstraints>
                <RowConstraints minHeight="10.0" prefHeight="30.0"
vgrow="SOMETIMES" />
                <RowConstraints minHeight="10.0" prefHeight="30.0"
vgrow="SOMETIMES" />
            </rowConstraints>
            <children>
                <Label text="מַטְבָּרָה :" GridPane.columnIndex="1" />
                <Label text="מַטְבָּרָה :" GridPane.columnIndex="1"
GridPane.rowIndex="1" />
                <Label text="לִזְעֵן :" GridPane.columnIndex="1"
GridPane.rowIndex="2" />
                <Label text="אַתְּ ?" GridPane.columnIndex="1"
GridPane.rowIndex="3" />
                <Label text="מִזְמְרָה :" GridPane.columnIndex="1"
GridPane.rowIndex="5" />
                <Label text="מִזְמְרָה :" GridPane.columnIndex="1"
GridPane.rowIndex="4" />
                <Label text="מִזְמְרָה :" GridPane.columnIndex="1"
GridPane.rowIndex="6" />
            </children>
        </GridPane>
    </center>

```

```

GridPane.rowIndex="7" />
    <Label text="טלפון : " GridPane.columnIndex="1"
GridPane.rowIndex="8" />
    <Label text="בנק מסגר : " GridPane.columnIndex="1"
GridPane.rowIndex="10" />
    <Label text="שם מסגר : " GridPane.columnIndex="1"
GridPane.rowIndex="11" />
    <TextField fx:id="tf_sneefBankCode" maxWidth="200.0"
prefHeight="25.0" prefWidth="627.0" GridPane.rowIndex="11" />
    <Label text="שם מסגר ליבורן" GridPane.columnIndex="1"
GridPane.rowIndex="12" />
    <TextField fx:id="tf_acountNumber" maxWidth="200.0"
prefHeight="25.0" prefWidth="627.0" GridPane.rowIndex="12" />
    <GridPane>
        <columnConstraints>
            <ColumnConstraints hgrow="SOMETIMES" maxWidth="450.0"
minWidth="10.0" prefWidth="450.0" />
            <ColumnConstraints hgrow="SOMETIMES" maxWidth="320.0"
minWidth="10.0" prefWidth="202.0" />
        </columnConstraints>
        <rowConstraints>
            <RowConstraints minHeight="10.0" prefHeight="30.0"
vgrow="SOMETIMES" />
        </rowConstraints>
        <children>
            <TextField fx:id="tf(firstName" alignment="CENTER_RIGHT"
maxWidth="200.0" prefHeight="25.0" prefWidth="627.0"
GridPane.columnIndex="1" GridPane.halignment="RIGHT" />
            <Label text="*" textFill="#f2290a"
GridPane.halignment="RIGHT">
                <font>
                    <Font size="18.0" />
                </font>
            </Label>
        </children>
    </GridPane>
<GridPane GridPane.rowIndex="1">
    <columnConstraints>
        <ColumnConstraints hgrow="SOMETIMES" maxWidth="451.0"
minWidth="10.0" prefWidth="451.0" />
        <ColumnConstraints hgrow="SOMETIMES" maxWidth="321.0"
minWidth="10.0" prefWidth="201.0" />
    </columnConstraints>
    <rowConstraints>
        <RowConstraints minHeight="10.0" prefHeight="30.0"
vgrow="SOMETIMES" />
    </rowConstraints>
    <children>
        <TextField fx:id="tf(lastName" maxWidth="200.0"
prefHeight="25.0" prefWidth="151.0" GridPane.columnIndex="1" />
        <Label text="*" textFill="#f2290a"
GridPane.halignment="RIGHT">
            <font>
                <Font size="18.0" />
            </font>
        </Label>
    </children>

```

```

minWidth="10.0" prefWidth="452.0" />
    <ColumnConstraints hgrow="SOMETIMES" maxWidth="321.0"
minWidth="10.0" prefWidth="200.0" />
</columnConstraints>
<rowConstraints>
    <RowConstraints minHeight="10.0" prefHeight="30.0"
vgrow="SOMETIMES" />
</rowConstraints>
<children>
    <TextField fx:id="tf_employeeID" maxWidth="200.0"
prefHeight="25.0" prefWidth="627.0" GridPane.columnIndex="1"
GridPane.halignment="RIGHT" />
        <Label text="*" textFill="#f2290a"
GridPane.halignment="RIGHT">
            <font>
                <Font size="18.0" />
            </font>
        </Label>
    </children>
</GridPane>
<GridPane GridPane.rowIndex="5">
    <columnConstraints>
        <ColumnConstraints hgrow="SOMETIMES" maxWidth="448.0"
minWidth="10.0" prefWidth="448.0" />
        <ColumnConstraints hgrow="SOMETIMES" maxWidth="319.0"
minWidth="10.0" prefWidth="204.0" />
    </columnConstraints>
    <rowConstraints>
        <RowConstraints minHeight="10.0" prefHeight="30.0"
vgrow="SOMETIMES" />
    </rowConstraints>
    <children>
        <Label text="*" textFill="#f2290a"
GridPane.halignment="RIGHT">
            <font>
                <Font size="18.0" />
            </font>
        </Label>
        <TextField fx:id="tf_phoneNumber" maxWidth="200.0"
prefHeight="25.0" prefWidth="627.0" GridPane.columnIndex="1" />
    </children>
</GridPane>
<ChoiceBox fx:id="cb_gender" minWidth="80.0" prefHeight="25.0"
prefWidth="52.0" GridPane.rowIndex="4">
    <items>
        <FXCollections fx:factory="observableArrayList">
            <String fx:value="ذكر" />
            <String fx:value="إناث" />
        </FXCollections>
    </items>
</ChoiceBox>
<GridPane GridPane.rowIndex="7">
    <columnConstraints>
        <ColumnConstraints hgrow="SOMETIMES" maxWidth="467.8"
minWidth="10.0" prefWidth="462.2000244140625" />
        <ColumnConstraints hgrow="SOMETIMES" maxWidth="200.0"
minWidth="10.0" prefWidth="101.59997558593744" />
        <ColumnConstraints hgrow="SOMETIMES" maxWidth="200.0"
minWidth="10.0" prefWidth="45.39998779296877" />
    </columnConstraints>
    <rowConstraints>

```

```

<RowConstraints minHeight="10.0" prefHeight="30.0"
vgrow="SOMETIMES" />
</rowConstraints>
<children>
    <TextField fx:id="tf_salaryPerHour" maxWidth="100.0"
prefHeight="25.0" prefWidth="627.0" GridPane.halignment="RIGHT" />
    <CheckBox fx:id="cb_isHourly" mnemonicParsing="false"
GridPane.columnIndex="2" GridPane.halignment="RIGHT" />
    <Label text="₪ תעריף שעה :" GridPane.columnIndex="1"
GridPane.halignment="LEFT" />
</children>
</GridPane>
<TextField fx:id="tf_bankNumber" maxWidth="200.0"
prefHeight="25.0" prefWidth="627.0" GridPane.rowIndex="10" />
<GridPane GridPane.rowIndex="13">
    <columnConstraints>
        <ColumnConstraints halignment="CENTER" hgrow="SOMETIMES"
minWidth="10.0" prefWidth="100.0" />
        <ColumnConstraints halignment="CENTER" hgrow="SOMETIMES"
minWidth="10.0" prefWidth="100.0" />
    </columnConstraints>
    <rowConstraints>
        <RowConstraints minHeight="10.0" prefHeight="30.0"
vgrow="SOMETIMES" />
    </rowConstraints>
</GridPane>
<Label text="טלפון סלולרי :" GridPane.columnIndex="1"
GridPane.halignment="CENTER" GridPane.rowIndex="9" />
<TextField fx:id="tf_salaryToTravel" maxWidth="200.0"
prefHeight="25.0" prefWidth="627.0" GridPane.halignment="RIGHT"
GridPane.rowIndex="9" />
<TextField fx:id="tf_email" maxWidth="200.0" prefHeight="25.0"
prefWidth="627.0" GridPane.rowIndex="3" />
<GridPane GridPane.rowIndex="6">
    <columnConstraints>
        <ColumnConstraints hgrow="SOMETIMES"
maxWidth="525.6000122070312" minWidth="10.0" prefWidth="525.6000122070312"
/>
        <ColumnConstraints hgrow="SOMETIMES"
maxWidth="319.60003662109375" minWidth="80.0"
prefWidth="127.19998779296873" />
    </columnConstraints>
    <rowConstraints>
        <RowConstraints minHeight="10.0" prefHeight="30.0"
vgrow="SOMETIMES" />
    </rowConstraints>
</GridPane>
<children>
    <ChoiceBox fx:id="cb_role" minWidth="80.0"
prefHeight="25.0" prefWidth="52.0" GridPane.columnIndex="1"
GridPane.halignment="RIGHT">
        <items>
            <FXCollections fx:factory="observableArrayList">
                <String fx:value="secretary" />
                <String fx:value="headOfDepartment" />
                <String fx:value="employee" />
            </FXCollections>
        </items>
    </ChoiceBox>
    <TextField fx:id="tf_password" alignment="CENTER_RIGHT"
disable="true" maxWidth="160.0" promptText="סיסמה בחר נא"
GridPane.halignment="RIGHT" />

```

```

                </children>
            </GridPane>
            <GridPane GridPane.rowIndex="8">
                <columnConstraints>
                    <ColumnConstraints hgrow="SOMETIMES"
maxWidth="590.399951171875" minWidth="10.0" prefWidth="534.3999755859375"
/>
                    <ColumnConstraints hgrow="SOMETIMES" maxWidth="200.0"
minWidth="10.0" prefWidth="534.3999755859375" />
                    <ColumnConstraints hgrow="SOMETIMES" maxWidth="200.0"
minWidth="10.0" prefWidth="117.60002441406255" />
                </columnConstraints>
                <rowConstraints>
                    <RowConstraints minHeight="10.0" prefHeight="30.0"
vgrow="SOMETIMES" />
                </rowConstraints>
                <children>
                    <CheckBox fx:id="cb_isPerMoth" mnemonicParsing="false"
GridPane.columnIndex="2" GridPane.halignment="RIGHT" />
                    <TextField fx:id="tf_perMonth" maxWidth="100.0"
GridPane.halignment="RIGHT" />
                    <Label text="רשות יפלון :" GridPane.columnIndex="1" />
                </children>
            </GridPane>
        </children>
    </GridPane>
    </center>
    <bottom>
        <ButtonBar prefHeight="40.0" prefWidth="200.0"
BorderPane.alignment="CENTER">
            <buttons>
                <Button maxHeight="40.0" maxWidth="150.0" minHeight="40.0"
minWidth="150.0" mnemonicParsing="false" onAction="#deletebtnclick"
text="הסרה">
                    <effect>
                        <DropShadow>
                            <color>
                                <Color red="0.02730000190734863"
green="0.24797500669956207" blue="0.9100000262260437" />
                            </color>
                        </DropShadow>
                    </effect>
                </Button>
                <Button fx:id="saveButton" maxHeight="40.0" maxWidth="150.0"
minHeight="40.0" minWidth="150.0" mnemonicParsing="false"
onAction="#saveBtnClicked" prefHeight="35.0" prefWidth="122.0" text="ירוחם"
">
                    <font>
                        <Font name="System Bold Italic" size="16.0" />
                    </font>
                    <effect>
                        <DropShadow>
                            <color>
                                <Color red="0.5473684072494507"
green="0.010136452503502369" blue="0.010136452503502369" />
                            </color>
                        </DropShadow>
                    </effect>
                </Button>
                <Button fx:id="backButton" maxHeight="40.0" maxWidth="150.0"
minHeight="40.0" minWidth="150.0" mnemonicParsing="false"

```

```

onAction="#backbtn" text="הנראן">
    <font>
        <Font name="System Bold Italic" size="16.0" />
    </font>
    <effect>
        <InnerShadow />
    </effect>
</Button>
</buttons>
<BorderPane.margin>
    <Insets />
</BorderPane.margin>
<padding>
    <Insets bottom="20.0" right="50.0" />
</padding>
</ButtonBar>
</bottom>
<top>
    <BorderPane maxHeight="150.0" minHeight="50.0" prefHeight="200.0"
prefWidth="200.0" BorderPane.alignment="CENTER">
        <top>
            <Label text="לכד ערך הסרת דבון" BorderPane.alignment="CENTER">
                <effect>
                    <InnerShadow>
                        <color>
                            <Color red="0.8421052694320679"
green="0.49278751015663147" blue="0.49278751015663147" />
                            </color>
                        </InnerShadow>
                    </effect>
                </Label>
            </top>
            <center>
                <GridPane alignment="TOP_RIGHT" maxHeight="40.0"
minHeight="40.0" BorderPane.alignment="CENTER">
                    <columnConstraints>
                        <ColumnConstraints halignment="RIGHT" hgrow="SOMETIMES"
maxWidth="822.3999633789062" minWidth="10.0" prefWidth="817.6000366210938"/>
                        <ColumnConstraints halignment="RIGHT" hgrow="SOMETIMES"
maxWidth="150.0" minWidth="150.0" prefWidth="158.39996337890625" />
                    </columnConstraints>
                    <rowConstraints>
                        <RowConstraints minHeight="10.0" prefHeight="30.0"
vgrow="SOMETIMES" />
                    </rowConstraints>
                <children>
                    <Label text="עוז בירת : " GridPane.columnIndex="1" />
                    <ChoiceBox fx:id="cb_retrieveEmployee" prefWidth="150.0">
                        <opaqueInsets>
                            <Insets />
                        </opaqueInsets>
                    </ChoiceBox>
                </children>
                <opaqueInsets>
                    <Insets />
                </opaqueInsets>
            <BorderPane.margin>
                <Insets right="70.0" />
            </BorderPane.margin>
        </GridPane>
    </top>
</BorderPane>

```

```
</center>
</BorderPane>
</top>
</BorderPane>

.root {
    -fx-background-color: #f0f0f0; /* Set the background color */
    -fx-font-family: "Arial"; /* Change the default font-family */
    -fx-font-size: 14px; /* Change the default font size */
}

/* Style for labels */
.label {
    -fx-font-size: 16px; /* Increase label font size */
    -fx-text-fill: #000000; /* Set label text color to black */
}

/* Style for text fields */
.text-field {
    -fx-font-size: 14px;
}

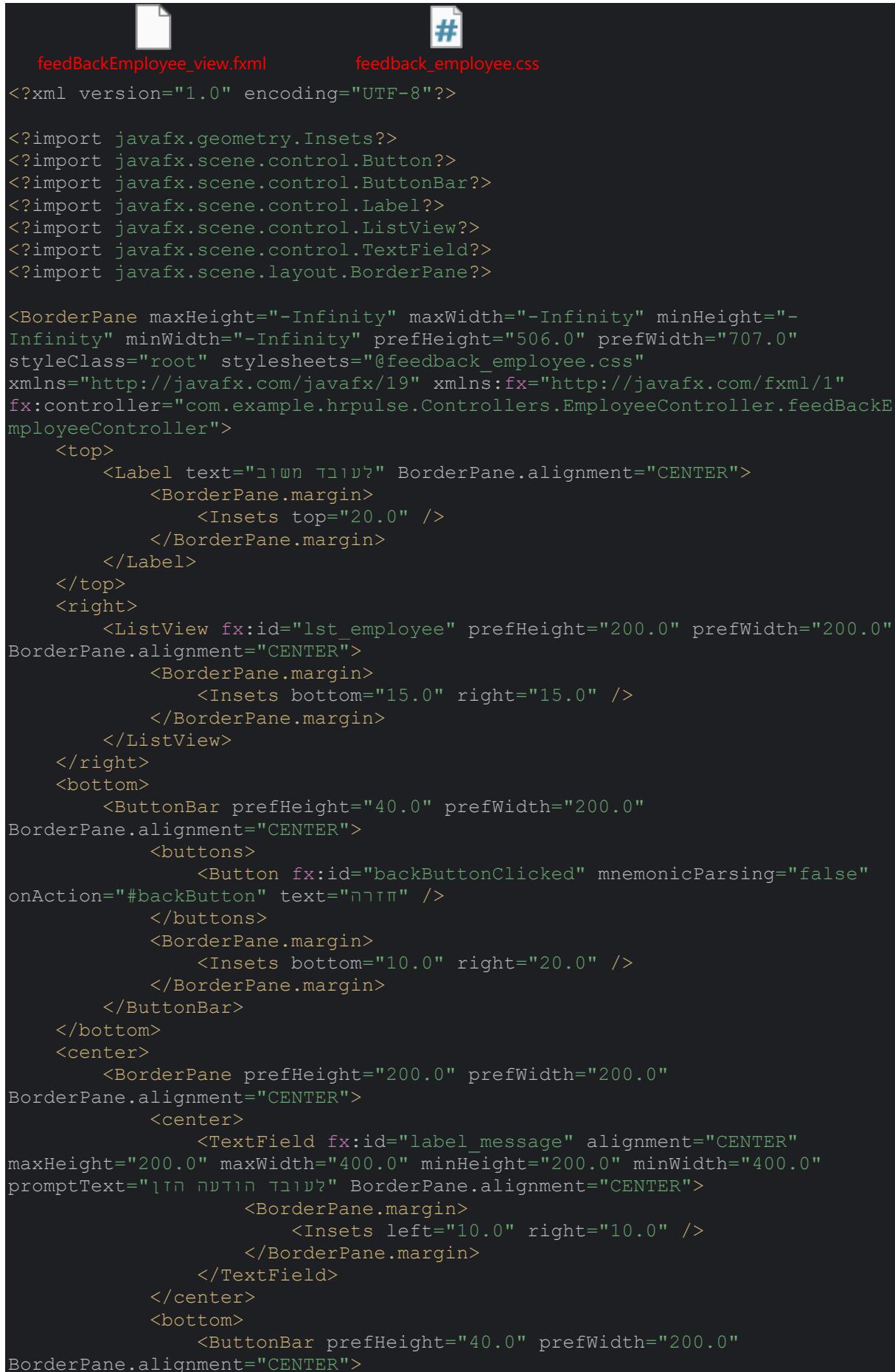
/* Style for choice boxes */
.choice-box {
    -fx-font-size: 14px;
}

/* Style for buttons */
.button {
    -fx-font-size: 16px;
    -fx-text-fill: #ffffff; /* Set button text color to white */
    -fx-background-color: #4CAF50; /* Default button background color */
}

/* Button hover effect */
.button:hover {
    -fx-background-color: #45a049; /* Button background color on hover */
    -fx-effect: innershadow(three-pass-box, #d8d8d8, 10, 0, 0, 0);
}

/* Style for date pickers */
.date-picker {
    -fx-font-size: 14px;
}

/* Style for checkboxes */
.checkbox {
    -fx-font-size: 14px;
}
```



The screenshot shows a file explorer window with two files listed:

- feedBackEmployee_view.fxml**: A FXML file icon.
- feedback_employee.css**: A CSS file icon.

The content of the **feedBackEmployee_view.fxml** file is as follows:

```
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.geometry.Insets?>
<?import javafx.scene.control.Button?>
<?import javafx.scene.control.ButtonBar?>
<?import javafx.scene.control.Label?>
<?import javafx.scene.control.ListView?>
<?import javafx.scene.control.TextField?>
<?import javafx.scene.layout.BorderPane?>

<BorderPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity" minWidth="-Infinity" prefHeight="506.0" prefWidth="707.0" styleClass="root" stylesheets="@feedback_employee.css" xmlns="http://javafx.com/javafx/19" xmlns:fx="http://javafx.com/fxml/1" fx:controller="com.example.hrpulse.Controllers.EmployeeController.feedBackEmployeeController">
    <top>
        <Label text="בבון מילוי" BorderPane.alignment="CENTER">
            <BorderPane.margin>
                <Insets top="20.0" />
            </BorderPane.margin>
        </Label>
    </top>
    <right>
        <ListView fx:id="lst_employee" prefHeight="200.0" prefWidth="200.0" BorderPane.alignment="CENTER">
            <BorderPane.margin>
                <Insets bottom="15.0" right="15.0" />
            </BorderPane.margin>
        </ListView>
    </right>
    <bottom>
        <ButtonBar prefHeight="40.0" prefWidth="200.0" BorderPane.alignment="CENTER">
            <buttons>
                <Button fx:id="backButtonClicked" mnemonicParsing="false" onAction="#backButton" text="חזור" />
            </buttons>
            <BorderPane.margin>
                <Insets bottom="10.0" right="20.0" />
            </BorderPane.margin>
        </ButtonBar>
    </bottom>
    <center>
        <BorderPane prefHeight="200.0" prefWidth="200.0" BorderPane.alignment="CENTER">
            <center>
                <TextField fx:id="label_message" alignment="CENTER" maxHeight="200.0" maxWidth="400.0" minHeight="200.0" minWidth="400.0" promptText="לין הגדת תכונת" BorderPane.alignment="CENTER">
                    <BorderPane.margin>
                        <Insets left="10.0" right="10.0" />
                    </BorderPane.margin>
                </TextField>
            </center>
            <bottom>
                <ButtonBar prefHeight="40.0" prefWidth="200.0" BorderPane.alignment="CENTER">
```

```

        <buttons>
            <Button fx:id="send_button" mnemonicParsing="false"
onAction="#sendButtonClicked" text="הנילש" />
        </buttons>
        <BorderPane.margin>
            <Insets right="40.0" />
        </BorderPane.margin>
    </ButtonBar>
</bottom>
</BorderPane>
</center>
</BorderPane>

.root {
    -fx-background-color: #f5f5f5; /* Light gray background color */
    /* You can add more global styling here */
}

#label_message {
    -fx-prompt-text-fill: #808080; /* Light gray prompt text */
}

#send_button, #backButtonClicked {
    -fx-background-color: #4CAF50; /* Green background color for buttons */
    -fx-text-fill: white; /* White text color */
    -fx-font-size: 14px; /* Font size */
}

#send_button:hover, #backButtonClicked:hover {
    -fx-background-color: #45a049; /* Darker green on hover */
}

```



manageEmployee_view.fxml



manageEmployeeStyle.css

```

<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.geometry.Insets?>
<?import javafx.scene.control.Button?>
<?import javafx.scene.control.ButtonBar?>
<?import javafx.scene.control.Label?>
<?import javafx.scene.layout.AnchorPane?>
<?import javafx.scene.layout.BorderPane?>
<?import javafx.scene.layout.ColumnConstraints?>
<?import javafx.scene.layout.GridPane?>
<?import javafx.scene.layout.RowConstraints?>
<?import javafx.scene.text.Font?>

<BorderPane prefHeight="594.0" prefWidth="818.0" styleClass="root"
styleSheets="@manageEmployeeStyle.css" xmlns="http://javafx.com/javafx/19"
xmlns:fx="http://javafx.com/fxml/1"
fx:controller="com.example.hrpulse.Controllers.EmployeeController.ManageEmp

```

```

loyeeController">
    <center>
        <AnchorPane prefHeight="102.0" prefWidth="142.0"
BorderPane.alignment="CENTER">
            <children>
                <GridPane layoutX="14.0" prefHeight="360.0" prefWidth="588.0"
AnchorPane.bottomAnchor="50.0" AnchorPane.leftAnchor="50.0"
AnchorPane.rightAnchor="50.0" AnchorPane.topAnchor="50.0">
                    <columnConstraints>
                        <ColumnConstraints halignment="RIGHT" hgrow="SOMETIMES"
maxWidth="539.0" minWidth="10.0" prefWidth="221.0" />
                        <ColumnConstraints halignment="RIGHT" hgrow="SOMETIMES"
maxWidth="539.0" minWidth="10.0" prefWidth="202.0" />
                    </columnConstraints>
                    <rowConstraints>
                        <RowConstraints minHeight="10.0" prefHeight="30.0"
vgrow="SOMETIMES" />
                        <RowConstraints minHeight="10.0" prefHeight="30.0"
vgrow="SOMETIMES" />
                    </rowConstraints>
                    <children>
                        <Button fx:id="addEmployeeButton" contentDisplay="RIGHT"
maxHeight="30.0" maxWidth="200.0" minHeight="30.0" minWidth="200.0"
mnemonicParsing="false" onAction="#addEmployeeClicked" text="הוסף ניובן"
GridPane.columnIndex="1">
                            <font>
                                <Font size="14.0" />
                            </font>
                        </Button>
                        <Button fx:id="editEmployeeButton" contentDisplay="RIGHT"
maxHeight="30.0" maxWidth="200.0" minHeight="30.0" minWidth="200.0"
mnemonicParsing="false" onAction="#editEmployeeClicked" text="ערוך ניובן /"
הסרת נדכון">
                            <font>
                                <Font size="14.0" />
                            </font>
                        </Button>
                        <Button fx:id="editShiftButton" contentDisplay="RIGHT"
maxHeight="30.0" maxWidth="200.0" minHeight="30.0" minWidth="200.0"
mnemonicParsing="false" onAction="#editShiftClicked" text="עדכון שעוררת נדכון"
GridPane.columnIndex="1" GridPane.rowIndex="1">
                            <font>
                                <Font size="14.0" />
                            </font>
                        </Button>
                        <Button fx:id="feedbackButton" contentDisplay="RIGHT"
maxHeight="30.0" maxWidth="200.0" minHeight="30.0" minWidth="200.0"
mnemonicParsing="false" onAction="#feedbackClicked" text="לעובן שונן נדכון"
GridPane.rowIndex="1">
                            <font>
                                <Font size="14.0" />
                            </font>
                        </Button>
                    </children>
                </GridPane>
            </children>
        </AnchorPane>
    </center>
    <bottom>
        <ButtonBar prefHeight="65.0" prefWidth="604.0"
BorderPane.alignment="CENTER">

```

```

<buttons>
    <Button fx:id="backButton" mnemonicParsing="false"
onAction="#backToManagerPage" text="חזור">
        <font>
            <Font size="14.0" />
        </font>
    </Button>
</buttons>
<BorderPane.margin>
    <Insets right="50.0" />
</BorderPane.margin>
</ButtonBar>
</bottom>
<top>
    <Label text="טבלה של עובדים" textFill="#b54f4f"
BorderPane.alignment="CENTER">
        <font>
            <Font size="18.0" />
        </font>
    </Label>
</top>
</BorderPane>

/* Root styles */
.root {
    -fx-background-color: white; /* Set the background color to white */
    -fx-background-size: cover;
}

/* Style for "הוסף עובד חדש" button */
#addEmployeeButton {
    -fx-background-color: #FF5733;
    -fx-text-fill: white;
    -fx-background-radius: 5;
    -fx-font-weight: bold;
}

/* Style for "הסרת עובד חדש" button */
#removeEmployeeButton {
    -fx-background-color: #3498DB;
    -fx-text-fill: white;
    -fx-background-radius: 5;
    -fx-font-weight: bold;
}

/* Style for "לעובי שמרת נתונים" button */
#editShiftButton {
    -fx-background-color: #E74C3C;
    -fx-text-fill: white;
    -fx-background-radius: 5;
    -fx-font-weight: bold;
}

/* Style for "לעובי נתונים" button */
#feedbackButton {
    -fx-background-color: #27AE60;
    -fx-text-fill: white;
    -fx-background-radius: 5;
}

```

```

        -fx-font-weight: bold;
    }

/* Style for "הציג מילוי שובע" button */
#employeeReportButton {
    -fx-background-color: #D35400;
    -fx-text-fill: white;
    -fx-background-radius: 5;
    -fx-font-weight: bold;
}

/* Style for "לתקן מילוי" button */
#updateRoleButton {
    -fx-background-color: #b54f4f;
    -fx-text-fill: white;
    -fx-background-radius: 5;
    -fx-font-weight: bold;
}

/* Style for the "חזרה" (back) button */
#backButton {
    -fx-background-color: #E67E22;
    -fx-text-fill: white;
    -fx-background-radius: 5;
    -fx-font-weight: bold;
}

/* Apply hover effect to all buttons */
.button:hover {
    -fx-background-color: derive(#34495E, 20%); /* Lighter background color
on hover */
    -fx-cursor: hand; /* Change cursor to a hand on hover */
    -fx-effect: dropshadow(three-pass-box, rgba(0, 0, 0, 0.3), 10, 0, 0,
0); /* Add a drop shadow on hover */
}

```



MonthlyShiftEmployee_report.fxml

```

<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.geometry.Insets?>
<?import javafx.scene.control.Button?>
<?import javafx.scene.control.ButtonBar?>
<?import javafx.scene.control.ChoiceBox?>
<?import javafx.scene.control.Label?>
<?import javafx.scene.control.ListView?>
<?import javafx.scene.layout.BorderPane?>
<?import javafx.scene.layout.ColumnConstraints?>
<?import javafx.scene.layout.GridPane?>
<?import javafx.scene.layout.HBox?>
<?import javafx.scene.layout.RowConstraints?>

<BorderPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-
Infinity" minWidth="-Infinity" nodeOrientation="RIGHT_TO_LEFT"
prefHeight="520.0" prefWidth="745.0" xmlns="http://javafx.com/javafx/21"
xmlns:fx="http://javafx.com/fxml/1"
fx:controller="com.example.hrpulse.Controllers.ReportsControllers.MonthlySh

```

```



```



```
    <ListView fx:id="lv_retrieveEmployees"
nodeOrientation="RIGHT_TO_LEFT" prefHeight="200.0" prefWidth="200.0"
BorderPane.alignment="CENTER" />

</center>
</BorderPane>
```

productionOfReports_view.fxml

```
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.geometry.Insets?>
<?import javafx.scene.control.Button?>
<?import javafx.scene.control.ButtonBar?>
<?import javafx.scene.control.Label?>
<?import javafx.scene.layout.BorderPane?>
<?import javafx.scene.layout.ColumnConstraints?>
<?import javafx.scene.layout.GridPane?>
<?import javafx.scene.layout.RowConstraints?>

<BorderPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-
Infinity" minWidth="-Infinity" prefHeight="400.0" prefWidth="600.0"
xmlns="http://javafx.com/javafx/21" xmlns:fx="http://javafx.com/fxml/1"
fx:controller="com.example.hrpulse.Controllers.ReportsControllers.productio
nOfReportsController">
    <top>
        <Label alignment="CENTER" contentDisplay="CENTER" maxHeight="20.0"
maxWidth="100.0" minHeight="20.0" minWidth="100.0" text="לקרן תוצאות"
BorderPane.alignment="CENTER" />
    </top>
    <bottom>
        <ButtonBar prefHeight="40.0" prefWidth="200.0"
BorderPane.alignment="CENTER">
            <buttons>
                <Button fx:id="back_btn" maxHeight="20.0" maxWidth="100.0"
minHeight="30.0" minWidth="100.0" mnemonicParsing="false"
onAction="#backToMainClicked" text="חזור" />
            </buttons>
            <BorderPane.margin>
                <Insets bottom="10.0" right="20.0" />
            </BorderPane.margin>
        </ButtonBar>
    </bottom>
    <center>
        <GridPane alignment="CENTER" nodeOrientation="RIGHT_TO_LEFT"
BorderPane.alignment="CENTER">
            <columnConstraints>
                <ColumnConstraints halignment="CENTER" hgrow="SOMETIMES"
minWidth="10.0" prefWidth="100.0" />
            </columnConstraints>
            <rowConstraints>
                <RowConstraints minHeight="10.0" prefHeight="30.0"
valignment="CENTER" vgrow="SOMETIMES" />
                <RowConstraints minHeight="10.0" prefHeight="30.0"
vgrow="SOMETIMES" />
            </rowConstraints>
            <children>
```

```

        <Button fx:id="employee_reports_btn" maxHeight="30.0"
maxWidth="175.0" minHeight="30.0" minWidth="150.0" mnemonicParsing="false"
onAction="#employeeReportsClicked" prefHeight="30.0" prefWidth="175.0"
text="מזהה הלקוח מפרט" />
        <Button fx:id="shiftEmployee_reports_btn" maxHeight="30.0"
maxWidth="150.0" minHeight="30.0" minWidth="150.0" mnemonicParsing="false"
onAction="#shiftEmployeeReportsClicked" text="מזהה שולחני דוחות"
GridPane.rowIndex="1" />
    </children>
    <padding>
        <Insets bottom="10.0" left="10.0" right="10.0" top="10.0"/>
    </padding>
</GridPane>
</center>
</BorderPane>

```



Users_view.fxml



UsersStyle.css

```

<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.geometry.Insets?>
<?import javafx.scene.control.Button?>
<?import javafx.scene.control.ButtonBar?>
<?import javafx.scene.control.Label?>
<?import javafx.scene.layout.BorderPane?>
<?import javafx.scene.layout.ColumnConstraints?>
<?import javafx.scene.layout.GridPane?>
<?import javafx.scene.layout.RowConstraints?>
<?import javafx.scene.text.Font?>

<BorderPane fx:id="borderPane" blendMode="DARKEN" prefHeight="652.0"
prefWidth="1002.0" styleClass="root" stylesheets="@UsersStyle.css"
xmlns="http://javafx.com/javafx/19" xmlns:fx="http://javafx.com/fxml/1"
fx:controller="com.example.hrpulse.Controllers.UserController">
    <center>
        <GridPane alignment="CENTER" prefHeight="318.0" prefWidth="434.0"
BorderPane.alignment="CENTER_LEFT">
            <columnConstraints>
                <ColumnConstraints halignment="RIGHT" hgrow="SOMETIMES" />
            </columnConstraints>
            <rowConstraints>
                <RowConstraints maxHeight="116.0000244140625" minHeight="10.0"
prefHeight="78.39998779296874" vgrow="SOMETIMES" />
                <RowConstraints maxHeight="164.80000610351564" minHeight="10.0"
prefHeight="92.40001220703121" vgrow="SOMETIMES" />
                <RowConstraints maxHeight="164.80000610351564" minHeight="10.0"
prefHeight="92.40001220703121" vgrow="SOMETIMES" />
                <RowConstraints maxHeight="164.80000610351564" minHeight="10.0"
prefHeight="92.40001220703121" vgrow="SOMETIMES" />
            </rowConstraints>
            <children>
                <Button fx:id="shiftHandlingbutton" alignment="CENTER"
mnemonicParsing="false" onAction="#manageEmployeeClicked" prefHeight="60.0"
prefWidth="120.0" text="מזהה ניחול">
                    <GridPane.margin>

```

```

            <Insets right="40.0" />
        </GridPane.margin></Button>
        <Button fx:id="reportsButton" alignment="CENTER"
mnemonicParsing="false" onAction="#NavigateToReportsPage" prefHeight="60.0"
prefWidth="252.0" text="נִיחוֹל דְּמָרֵכֶת וּבְקַרְבָּן (Future Feature)"
GridPane.rowIndex="2">
            <GridPane.margin>
                <Insets right="40.0" />
            </GridPane.margin></Button>
        <Button fx:id="production_of_report_btn" alignment="CENTER"
mnemonicParsing="false" onAction="#reportsClicked" prefHeight="60.0"
prefWidth="120.0" text="תְּקִפְתָּה לְמִינְיָן" GridPane.rowIndex="3">
            <GridPane.margin>
                <Insets right="40.0" />
            </GridPane.margin></Button>
        <Button fx:id="EditEmployeeButton" alignment="CENTER"
mnemonicParsing="false" onAction="#NavigateToManageDepartment"
prefHeight="60.0" prefWidth="120.0" text="מִנהל קְרָבָּת וּנִיחוֹל"
GridPane.rowIndex="1">
            <GridPane.margin>
                <Insets right="40.0" />
            </GridPane.margin></Button>
        </children>
    <padding>
        <Insets left="200.0" />
    </padding>
    <opaqueInsets>
        <Insets />
    </opaqueInsets>
</GridPane>
</center>
<bottom>
    <ButtonBar nodeOrientation="LEFT_TO_RIGHT" prefHeight="40.0"
prefWidth="200.0" BorderPane.alignment="TOP_LEFT">
        <buttons>
            <Label fx:id="label_userName" nodeOrientation="LEFT_TO_RIGHT"
text="דָּעַת שְׁמוֹן :" textFill="#e7e1e1">
                <opaqueInsets>
                    <Insets right="1000.0" />
                </opaqueInsets>
                <font>
                    <Font size="16.0" />
                </font>
            </Label>
            <Button fx:id="homePageButton" alignment="CENTER"
contentDisplay="RIGHT" mnemonicParsing="false" onAction="#HomePageClicked"
prefHeight="60.0" prefWidth="120.0" text="בָּאָמָן שְׁמוֹן" />
            <Button fx:id="exitButton" alignment="CENTER"
mnemonicParsing="false" onAction="#ExitButtonClicked" prefHeight="60.0"
prefWidth="120.0" text="הַרְצָן" />
        </buttons>
    <padding>
        <Insets bottom="20.0" right="40.0" />
    </padding>
</ButtonBar>
</bottom>
</BorderPane>

```

```

/* Root styles */
.root.manager{
    -fx-background-image: url("Images/manager1.png");
    -fx-background-size: cover;
}
.root.secretary{
    -fx-background-image: url("Images/secretary2.png");
    -fx-background-size: cover;
}
.root.headOfDepartment{
    -fx-background-image: url("Images/headOfDepartment.jpeg");
    -fx-background-size: cover;
}

/* Style for "מחקרים נייחות" button */
#EditEmployeeButton {
    -fx-background-color: #FF5733; /* Set the background color */
    -fx-text-fill: white; /* Set the text color */
    -fx-background-radius: 5; /* Rounded corners */
    -fx-font-weight: bold;
}

/* Style for "טובדים נייחות" button */
#shiftHandlingbutton {
    -fx-background-color: #3498DB;
    -fx-text-fill: white;
    -fx-background-radius: 5;
    -fx-font-weight: bold;
}

/* Style for "ובקשות דוחות נייחות" button */
#reportsButton {
    -fx-background-color: #E74C3C;
    -fx-text-fill: white;
    -fx-background-radius: 5;
    -fx-font-weight: bold;
}

/* Style for "מיפוי הפקה" button */
#production_of_report_btn {
    -fx-background-color: #27AE60;
    -fx-text-fill: white;
    -fx-background-radius: 5;
    -fx-font-weight: bold;
}

/* Apply hover effect to all buttons */
.button:hover {
    -fx-background-color: derive(#34495E, 20%); /* Lighter background color on hover */
    -fx-cursor: hand; /* Change cursor to a hand on hover */
    -fx-effect: dropshadow(three-pass-box, rgba(0, 0, 0, 0.3), 10, 0, 0, 0); /* Add a drop shadow on hover */
}
/* Style for the "דף הבית" and "יציאה" buttons in the bottom ButtonBar */
*/
#homePageButton, #exitButton {
    -fx-background-color: #E67E22;
    -fx-text-fill: white;
    -fx-background-radius: 5;
    -fx-font-weight: bold;
}

```

```
}

/* Apply hover effect to the "דף הבית" and "יציאה" buttons */
#homePageButton:hover, #exitButton:hover {
    -fx-background-color: #D35400;
    -fx-cursor: hand;
}
.button {
    -fx-font-family: "Arial"; /* Change font family */
    -fx-font-size: 14px; /* Change font size */
    -fx-text-fill: white; /* Set text color to white */

    -fx-min-width: 150px;
    -fx-min-height: 40px;
}
.button:hover .text {
    -fx-fill: lightgrey; /* Change font color to light grey on hover */
}
```