A black text on a white background

Description automatically generated

תוכנית הלימודים לתואר ראשון בהנדסת תוכנה

**שם הפרויקט: מערכת להקצאת מקומות ישיבה באולם קולנוע**

ספר הפרויקט

**שם הסטודנטית: שחר גולדברגר**

**שם הסטודנט: עמית כהן**

אישור המנחה על בדיקת המסמך ואישור להגשה:

**תאריך הגשה:**

**8.8.2024**

תוכן עניינים-

|  |  |
| --- | --- |
| היענות ועמידה בתכנון למול מסמכי הייזום והתכן/דוח הנדסי.................................................................... | 3 |
| אופן ההתייחסות למטרות והדרישות שהוגדרו במסמך הייזום והדוח ההנדסי................................................ | 3 |
| בעיות המעכבות את התקדמות הפרויקט............................................................................................... | 10 |
| עמידה או אי עמידה בGnatt תוכנית העבודה, מימוש וטיפול בסיכונים ומגבלות שהוגדרו............................... | 10 |
| סטייה מתאריך היעד של הגשת טיוטת ספר הפרויקט.............................................................................. | 13 |
| ניהול הסיכונים, למול המסמך הייזום והדו"ח ההנדסי............................................................................... | 14 |
| אופן ההתמודדות עם המגבלות שהוגדרו במסמכי התכן הקודמים.............................................................. | 18 |
| מפרטי תכן וארכיטקטורת המערכת שמומשה במסגרת הפרויקט............................................................... | 19 |
| תרשים ארכיטקטורה מפורט................................................................................................................. | 19 |
| התייחסות מפורטת לכל בלוק ומודל....................................................................................................... | 20 |
| מפרטי אלגוריתמים............................................................................................................................. | 24 |
| פירוט תצורות ותכולות מעטפת של התוצרים, המודולים והמערכת............................................................ | 31 |
| שיטות וכלים/פלטפורמות שמומשו במסגרת תכן ופיתוח המערכת............................................................ | 61 |
| מפרטי תכן וארכיטקטורת מערכת הAlpha שמומשה במסגרת הפרויקט..................................................... | 64 |
| תכן מפורט (Low Level) של וארכיטקטורת מערכת הAlpha שיושמה........................................................ | 64 |
| תרשים ארכיטקטורה מפורט ברמת בלוקים ומודולים טכנולוגיים של ה Alpha שמומשה במסגרת הפרויקט...... | 68 |
| הבלוקים והמודולים וכיצד משתלבים בתפקוד המערכת ובתרומה ליישום הארכיטקטורה הכוללת של המערכת | 69 |
| ממצאי ביצוע בדיקות מערכתיות........................................................................................................... | 71 |
| פירוט של ביצוע תוכנית הבדיקות של המערכת וה Alpha, על פי מסגרת התכנון שפורטה בדוח ההנדסי......... | 71 |
| דיון (Discussion)............................................................................................................................... | 73 |
| סיכום המסקנות והלקחים שהופקו....................................................................................................... | 73 |
| הצעה לעבודת המשך, על בסיס הפרויקט והמערכת שמומשו.................................................................... | 74 |
| סיכום ומסקנות (Summary and conclusions)..................................................................................... | 75 |
| מקורות מעודכנים (References).......................................................................................................... | 75 |
| נספחים (Appendices)...................................................................................................................... | 80 |
| נספח פוסטר הפרויקט........................................................................................................................ | 80 |
| מידע המסייע לתיאור הפרויקט ולהבנתו................................................................................................. | 81 |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**1 היענות ועמידה בתכנון למול מסמכי הייזום והתכן/דוח הנדסי**

**1.1 + 1.2 אופן ההתייחסות למטרות והדרישות שהוגדרו במסמך הייזום והדוח ההנדסי**

סעיף זה מספק סקירה מקיפה של אופן ההתייחסות למטרות והדרישות שהוגדרו במסמך הייזום ובדוח ההנדסי. הוא כולל תיאורים מפורטים של עמידה של הפרויקט במפרטים ראשוניים אלה ומדגיש את כל השינויים שחלו מאז הגשת המסמכים.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| מסמך | מטרה/ יעד מקוריים | מידת היענות למטרה | שינויים שבוצעו | משמעות השינוי |
| מסמך ייזום | בניית אלגוריתם המותאם לתצורות אולם שונות | המטרה של פיתוח אלגוריתם המסוגל להסתגל לתצורות אולמות שונות התקבלה בהצלחה מלאה. האלגוריתם תוכנן כדי להתאים לפריסות וסידורי הישיבה של אולמות קולנוע שונים, מה שמבטיח גמישות וחוסן ביישום שלו. | לאורך תהליך הפיתוח, שולבו מספר שיפורים באלגוריתם על מנת להבטיח ביצועים מיטביים ושביעות רצון המשתמשים. שינויים אלו כללו:  גמישות משופרת: בתחילה, האלגוריתם תוכנן עבור תצורות אולם סטנדרטיות. לאחר מכן, האלגוריתם שוכלל לטיפול במגוון רחב יותר של סידורי ישיבה, לא בהכרח סידורים סטנדרטיים של אולם (למשל- מטריצה NXN).כלומר, אפשר ליצור אירוע בתצורות שונות- מלבן, ריבוע, מעוין וכל צורה אחרת. כאשר רוצים ליצור אירוע חדש ויש להגדיר את מפת האירוע, יש אפשרות לתמחר כיסאות במחירים שונים. במידה וכיסא תומחר במחיר 0 ש"ח ע"י המשתמש, המשמעות היא שעבור כיסא זה האלגוריתם לא ינסה למצוא סידור ישיבה מתאים ולכן, למעשה, הכיסא לא נכלל בתור כיסא כלל במפת האולם.  החלק הפרויקט בו בוצע השינוי- תוצרים (הקוד).  מועד התרחשותו- 19.6.2024  יוזם/יוזמת השינוי- שחר גולדברגר ועמית כהן  אושר ע"י- ד"ר אלי וינטראוב. | לשינויים שנעשו באלגוריתם יש השלכות גדולות על יעילות המערכת:  הגמישות מבטיחה שניתן לפרוס את פריסות ההושבה במגוון רחב יותר של אולמות קולנוע, מה שהופך את המערכת לפתרון רב-תכליתי. יכולת הסתגלות זו מהווה אבן יסוד במערכת, שכן היא מאפשרת אינטגרציה חלקה עם תשתיות של אולמות קולנוע שונים. |
| מסמך ייזום | מקסום התפוסה באולם. | המטרה של מיקסום התפוסה באולם הושגה בדרגה גבוהה של הצלחה. המערכת תוכננה לייעל את הקצאת המושבים, תוך התחשבות באילוצים שונים כגון הנחיות ריחוק חברתי, העדפות משתמש ותצורות ישיבה. האלגוריתם המתקדם שפותח למטרה זו מבטיח שהמספר המרבי של מושבים מנוצל ביעילות (תוך שימוש בפונקציית מטרה שמטרתה להוביל למיקסום התפוסה באולם ומקסום הרווחים, הפונקציה עושה זאת ע"י פתרון אופטימלי והיוריסטיקה, מוסבר בהמשך), ובכך ממקסם את התפוסה תוך הקפדה על תקנות הבריאות. | במהלך הפרויקט, שולבו מספר שיפורים כדי לשפר את יעילות האלגוריתם במיקסום תפוסת האולם:  אם משתמש הכניס קלט שאין באפשרות המערכת למצוא עבורו סידור ישיבה, המערכת נבנתה בצורה אינטראקטיבית כך שתשאל את המשתמש האם ברצונו לקבל סידור שונה מהציפייה הראשונית שלו. במידה וכן, המערכת מציעה למשתמש הצעה חלופית והמשתמש בוחר האם ברצונו לקבלה. האלגוריתם השתמש בהיוריסטיקה למיטוב הפתרון, יש על כך פירוט בהמשך.  החלק הפרויקט בו בוצע השינוי- תוצרים (הקוד).  מועד התרחשותו- 19.6.2024  יוזם/יוזמת השינוי- שחר גולדברגר ועמית כהן.  אושר ע"י- ד"ר אלי וינטראוב. | השינויים שיושמו השפיעו באופן משמעותי על יכולת המערכת למקסם את התפוסה:  יעילות: יכולות ההתאמה הדינמיות מבטיחות שהמערכת יכולה להסתגל במהירות להנחיות חדשות או לשינויים בהתנהגות המשתמש, תוך שמירה על שיעורי תפוסה גבוהים גם בתנאים משתנים.  שינויים אלו הביאו למערכת חזקה, ניתנת להתאמה וידידותית למשתמש הממקסמת את התפוסה ביעילות תוך הבטחת הבטיחות והנוחות של צופי הסרטים. |
| מסמך ייזום | זמני תגובה מהירים עבור המשתמשים בזמן השימוש במערכת. | המטרה של השגת זמני תגובה מהירים למשתמשים בזמן השימוש במערכת הושגה בהצלחה. המערכת תוכננה בקפידה כדי להבטיח ועיבוד מהיר של בחירת המושבים והקצאת המושבים. על ידי אופטימיזציה של תהליכי קצה, המערכת מספקת חווית משתמש חלקה ומהירה. | במהלך תהליך הפיתוח, יושמו מספר שיפורים מרכזיים כדי להשיג ולשמור על זמני תגובה מהירים:  ארכיטקטורת השרת הראשונית ושאילתות מסד הנתונים כוונו ליעילות ביצועים. זה כלל אופטימיזציה של מסד הנתונים, טיפול בשאילתות ואיזון עומסים כדי להפחית את זמני התגובה.  יעילות קוד: בדיקות קוד ושינויים מתמשכים נערכו כדי להבטיח שבסיס הקוד נשאר רזה ויעיל, תוך צמצום זמני הביצוע ושיפור הביצועים הכוללים.  בנוסף, הפתרון החמדני משתמש כקלט לפתרון היוריסטי והחיפוש היוריסטי מחפש עד מרחק 10 כיסאות מהפתרון החמדני. יש לכך משמעות בזמני התגובה של המערכת, בעיקר עבור אולמות גדולים בהם מספר מקומות הישיבה גדול.  החלק הפרויקט בו בוצע השינוי- תוצרים (הקוד).  מועד התרחשותו- 21.6.2024  יוזם/יוזמת השינוי- שחר גולדברגר ועמית כהן.  אושר ע"י- ד"ר אלי וינטראוב. | חווית משתמש משופרת: זמני תגובה מהירים מבטיחים שמשתמשים יחוו זמני המתנה מינימליים בעת אינטראקציה עם המערכת, מה שמוביל לחוויה מהנה וחלקה יותר.  מעורבות מוגברת של משתמשים: אינטראקציות מהירות ומגיבות מעודדות משתמשים לעסוק בתדירות גבוהה יותר עם המערכת, מה שמגדיל את השימוש הכולל ואת שביעות רצון המשתמשים.  התשתית הניתנת להרחבה ותהליכי הקצה האחורי האופטימליים מבטיחים שהמערכת תישאר אמינה ומתפקדת היטב גם בתנאי שימוש שיא, ומונעים האטות או קריסות. |
| מסמך ייזום | שמירה על פרטיות המשתמשים ואבטחת מידע | המטרה של שמירה על פרטיות המשתמש ואבטחת המידע הושגה חלקית. בעוד שהושגה התקדמות משמעותית ביישום הרשאות עבור כל משתמש והבטחת שמשתמשים יוכלו לבצע פעולות בהתאם להגדרות שלהם, היבטים מסוימים של פרטיות ואבטחה לא עמדו במלואם ביעדים הראשוניים. תאימות חלקית זו מצביעה על כך שבעוד אלמנטים בסיסיים יושמו בהצלחה, יש צורך בשכלולים נוספים למימוש מלא של תקני האבטחה המיועדים.  מכוון שעיקר הפרויקט עוסק באלגוריתם הנבנה, לא ראינו לנכון להשקיע את רוב הזמן על הטמעת אלגוריתמי הצפנה, שכן, זה לא מהות הפרויקט. לכן, רוב המאמצים הושקעו בפיתוח אלגוריתם חזק ויעיל. | ננקטו מספר צעדים מרכזיים כדי לשפר את פרטיות המשתמש ואבטחת המידע, תוך התמקדות מיוחדת בהרשאות ופעולות המשתמש:  הרשאות משתמש: פותחה מערכת הרשאות חזקה, המאפשרת לבצע פעולות ספציפיות בהתבסס על תפקידי משתמש והגדרות. זה כלל הבחנה בין רמות גישה שונות, כגון מנהלי מערכת, משתמשים רגילים ואורחים.  בקרות גישה לנתונים: מנגנוני בקרת גישה ראשוניים הוקמו כדי להבטיח שמשתמשים יוכלו לגשת רק לנתונים הרלוונטיים להרשאות שלהם. עם זאת, בקרות אלה דורשות חיזוק נוסף כדי למנוע פרצות אבטחה אפשריות.  עם זאת, המערכת לא השתמשה באלגוריתמי הצפנה כדי להגן על פרטיות המידע, לכן, המטרה על פרטיות המשתמשים הושגה באופן חלקי.  מועד התרחשותו- 1.5.2024  יוזם/יוזמת השינוי- שחר גולדברגר ועמית כהן.  אושר ע"י- ד"ר אלי וינטראוב. | לשינויים שבוצעו יש כמה השלכות חשובות על מסגרת האבטחה והפרטיות הכוללת של המערכת:  הצלחה חלקית בהרשאות משתמש: הטמעה של מערכת הרשאות מאפשרת אינטראקציה מאובטחת ומאורגנת יותר של המשתמש בתוך הפלטפורמה. משתמשים יכולים לבצע פעולות המתאימות לתפקידיהם. |
| דו"ח הנדסי | ניצול אופטימלי של אולם בהתבסס על אלגוריתם אופטימיזציה גלובלי. | המטרה של השגת ניצול מיטבי של האולם על בסיס אלגוריתם אופטימיזציה גלובלי הושגה בהצלחה באמצעות גישה חדשנית ששונה מהתוכנית הראשונית. בתחילה נועד האלגוריתם למצוא סידור חמדני לכל הזמנה ובהמשך סידור גלובלי לכל האולם. הגישה המתוקנת כרוכה כעת במציאת סידור חמדני לכל הזמנה ולאחריה סידור ישיבה אופטימלי באופן מיידי תוך שימוש בהיוריסטיקה מבוססת פונקציית מטרה. תהליך זה מוחל על כל הזמנה בנפרד במקום לצבור את כל ההזמנות בבת אחת. | במקום לצבור את כל ההזמנות ולמצוא סידור גלובלי, האלגוריתם מייעל כעת כל הזמנה כפי שהיא מתקבלת. הדבר מבטיח שכל הזמנה מטופלת בצורה מיטבית בזמן אמת, מה שמוביל להקצאת מושבים יעילה יותר.  שילוב חמדני ואופטימלי: השלב הראשוני של מציאת סידור חמדן נשאר, אך מיד אחריו מגיע סידור אופטימלי לסדר הספציפי. גישה היברידית זו ממנפת את המהירות של אלגוריתמים חמדניים ואת היסודיות של אלגוריתמי אופטימיזציה.  שיפור מתמיד: האלגוריתם משכלל כעת את סידור הישיבה בכל הזמנה חדשה, ומבטיח אופטימיזציה מתמשכת והתאמה לתנאים משתנים ולהעדפות המשתמש.  מועד התרחשותו- 20.6.2024  יוזם/יוזמת השינוי- ד"ר אלי וינטראוב.  אושר ע"י- ד"ר אלי וינטראוב. | לשינויים שבוצעו יש השלכות עמוקות על יעילות המערכת ויעילותה:  אופטימיזציה בזמן אמת תוך שימוש בהיוריסטיקה: על ידי אופטימיזציה של כל הזמנה כשהיא נכנסת, המערכת מבטיחה שהקצאת המושבים תהיה תמיד אופטימלית, משפרת את הניצול הכולל של האולם ומשפרת את שביעות רצון הלקוחות.  השילוב של סידורים חמדניים ואופטימליים מאפשר הקצאת מושבים מהירה אך יעילה. גישה זו מפחיתה את זמן העיבוד ומבטיחה שהמערכת תוכל לטפל בהיקפים גבוהים של הזמנות ללא דחייה.  אופטימיזציה מתמשכת פירושה שהמערכת יכולה להסתגל לשינויים בזמן אמת בדפוסי הסדר ובהעדפות המשתמש, מה שמבטיח שהאולם מנוצל במלוא הפוטנציאל שלו בכל עת.  שביעות רצון הלקוחות: בניגוד לאלגוריתם הראשוני שתוכנן, כעת הלקוח מקבל ברגע ההזמנה את סידור הישיבה שהמערכת מצאה ולא לאחר זמן מה. יש על כך חשיבות |

**1.2.3 בעיות המעכבות את התקדמות הפרויקט-**

השפעת המלחמה-

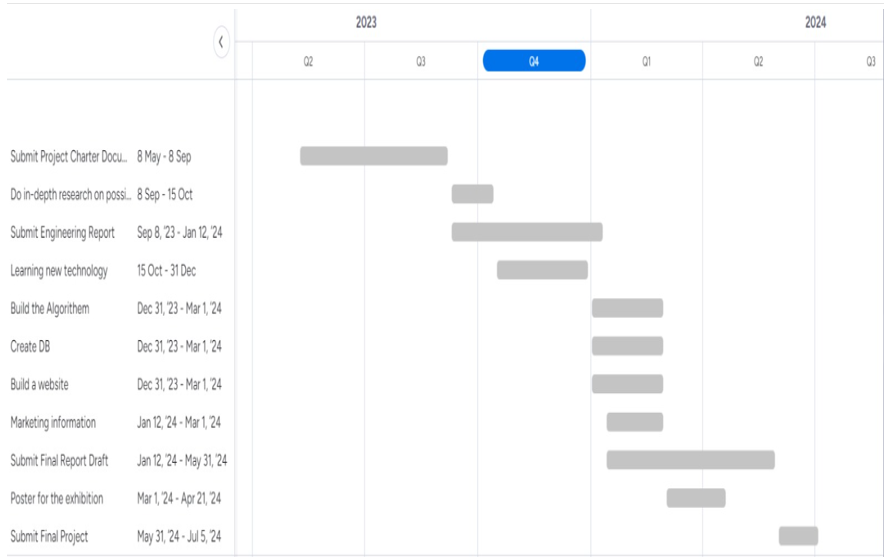
הפרויקט פותח בזמן מלחמה והדבר הכניס קשיים נוספים:

אחד מחברי הצוות זומן לעתים קרובות למילואים. הדבר יצר שיבושים בזרימת העבודה והאט את ההתקדמות של משימות פיתוח קריטיות. היעדר חבר צוות כה חיוני הצריך התאמות בלוחות הזמנים של הפרויקט וחלוקה מחדש של אחריות. נדגיש כי למרות העיכובים, אנחנו צפויים להגיש בזמן את פרויקט הגמר ולעמוד ברוב הדרישות שהצבנו.

בנוסף, למידה אודות מיטוב פתרונות ופתרונות היוריסטיים הצריך מאיתנו ללמוד אודותם, הדבר גזל מאיתנו זמן רב ועיכב את התקדמות הפרויקט, במיוחד בשלביו המתקדמים.

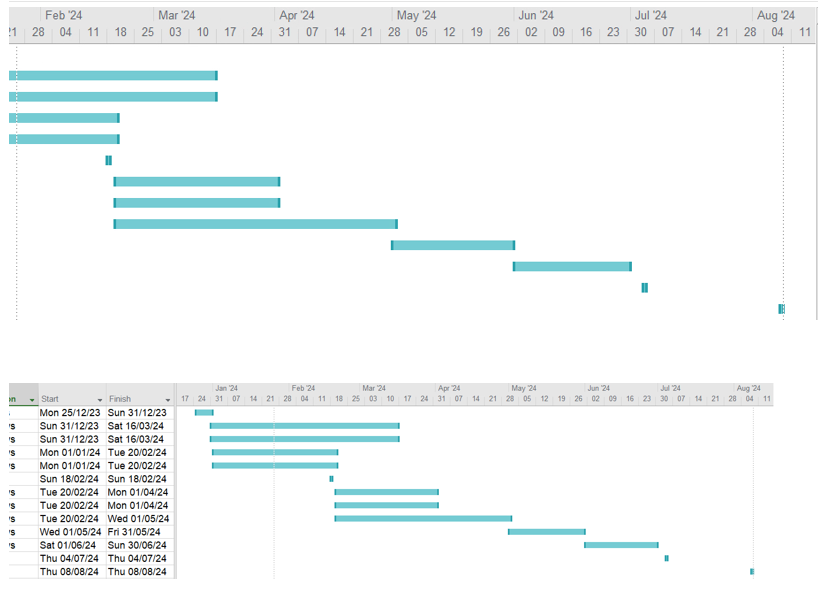
**1.3 עמידה / אי עמידה בGnatt תוכנית העבודה, מימוש וטיפול בסיכונים ומגבלות שהוגדרו-**

תרשים Gnatt מסמך ייזום (הוגש ב19.11.2023)



|  |  |  |
| --- | --- | --- |
| תאריך יעד | אחריות | מטלה |
| 26.01.2023 | צוות | בחירה ורישום לפרויקט גמר |
| 19.11.2023 | צוות | הגשת מסמך ייזום |
| 01.12.2023 | צוות | בחירת שפת התכנות איתה נעבוד- Python/Java/C++/JavaScript. שפות ליבה הנמצאות בשימוש נפוץ בפיתוח אלגוריתמי בשל הספריות והמסגרות החזקות שלהן התומכות במבני נתונים שונים, אלגוריתמים ושילובי אינטרנט. |
| 10.12.2023 | שחר | למידה עמוקה של SQL/NoSQL |
| 20.12.2023 | עמית | רכישת ידע באלגוריתמים מתקדמים: הבנה של אלגוריתמים חמדניים, תכנות דינמי, אלגוריתמי גרפים וכו', שיכולים לסייע בתכנון הלוגיקה של האלגוריתם. |
| 30.12.2023 | צוות | למידה של Git: חיוני לניהול קוד מקור. יאפשר לנו עבודה על הקוד במקביל. |
| 12.01.2013 | צוות | הגשת דוח הנדסי |
| 05.07.2023 | צוות | הגשת דוח סופי |
| מאי/יוני 2024 (טרם נקבע) | צוות | תערוכת פרויקטים |

תרשים Gnatt מסמך הנדסי (הוגש ב26.01.2024)



A screenshot of a calendar

Description automatically generated

|  |  |  |
| --- | --- | --- |
| הגדרת סביבת פיתוח Agile | 25.12.23 | שחר |
| למידה עמוקה של אלגוריתמי אופטימיזציה | 31.12.23 | שחר + עמית |
| למידה עמוקה של היוריסטיות | 31.12.23 | שחר + עמית |
| יצירת עיצוב וארכיטקטורה ראשוניים | 1.1.24 | שחר + עמית |
| עיצוב מסד נתונים | 1.1.24 | שחר |
| הגשת דוח הנדסי | 18.2.24 | שחר + עמית |
| יצירת UI/UX | 20.2.24 | עמית |
| פיתוח ממשק Frontend (ממשק הזמנות) | 20.2.24 | עמית |
| פיתוח אלגוריתם הקצאת המושבים | 20.2.24 | שחר+ עמית |
| תערוכת פרויקטים | מאי/יוני 2024 (טרם נקבע) | שחר + עמית |
| תיעוד (באמצעות Git ) | באופן תדיר | שחר + עמית |
| הגשת טיוטת ספר פרויקט | 4.7.24 | שחר + עמית |
| הגשת קובץ סופי | 8.8.24 | שחר + עמית |
| בחינה על פרויקט הגמר | 9.9.2024 | שחר + עמית |

לוח זמנים משוכלל- מסמך ייזום + מסמך הנדסי-

|  |  |  |
| --- | --- | --- |
| משימה | תאריך יעד (ירוק- עמידה ביעד, אדום- אי עמידה ביעד, לא מסומן- עדיין לא הגענו לתאריך) | אחריות |
| בחירה ורישום לפרויקט גמר | 26.01.2023 | שחר + עמית |
| הגשת מסמך ייזום | 19.11.2024 | שחר + עמית |
| בחירת שפת התכנות איתה נעבוד | 1.12.2024 | שחר + עמית |
| למידה עמוקה של SQL/NoSQL | 10.12.2023 | שחר |
| למידה עמוקה של אלגוריתמי אופטימיזציה | 20.12.2023 | עמית |
| למידה עמוקה של היוריסטיות | 31.12.23 | שחר + עמית |
| למידה של Git | 30.12.2024 |  |
| יצירת עיצוב וארכיטקטורה ראשוניים | 1.1.24 | שחר + עמית |
| עיצוב מסד נתונים | 1.1.24 | שחר |
| הגשת דוח הנדסי | 18.2.24 | שחר + עמית |
| יצירת UI/UX | 20.2.24 | עמית |
| פיתוח ממשק Frontend (ממשק הזמנות) | 20.2.24 | עמית |
| פיתוח אלגוריתם הקצאת המושבים | 20.2.24 | שחר+ עמית |
| תערוכת פרויקטים | טרם נקבע | שחר + עמית |
| תיעוד (באמצעות Git ) | באופן תדיר | שחר + עמית |
| הגשת טיוטת ספר פרויקט | 4.7.24 תאריך מעודכן 25.7.2 (מאושר ע"י ניר) | שחר + עמית |
| הגשת קובץ סופי- ספר הפרויקט | 8.8.24 | שחר + עמית |
| בחינה על הפרויקט הגמר | 9.9.2024 | שחר + עמית |

**1.3.5 סטייה מתאריך היעד של הגשת טיוטת ספר הפרויקט-**

הגשת טיוטת ספר הפרויקט התעכבה מאחר ועמית נקרא למילואים לסירוגין ולא הספקנו להשלים את כתיבת הספר בזמן. לקראת ההגשה (כחודש לפני), פנינו לניר שאישר לנו להגיש את טיוטת הספר האיחור של 3 שבועות מתאריך היעד. השפעת העיכוב הייתה מינורית מאחר ורוב הקוד של המערכת נכתב והיינו צריכים בעיקר זמן כדי לכתוב את טיוטת ספר הפרויקט, טיוטת ספר הפרויקט הוגשה כ3 שבועות מהתאריך הסופי.

את ספר הפרויקט אנחנו צפויים להגיש בזמן עד התאריך 8.8.24, זאת למרות שניתנה לנו הארכה נוספת של שבוע.

**1.3.6 ניהול הסיכונים, למול המסמך הייזום והדו"ח ההנדסי-**

פרק זה מתאר את תכנית ניהול הסיכונים כמפורט במסמכי הייזום ובדוח ההנדסי. הפרק מספק דגשים וניתוח מפורט של הסיכונים שהתממשו, הטיפול בהם והשפעתם על ההתקדמות והיישום של הפרויקט. ניהול סיכונים יעיל כולל זיהוי סיכונים פוטנציאליים, הערכת השפעתם והסבירות שלהם ויישום אסטרטגיות להפחתתם. הסעיפים הבאים מפרטים את הסיכונים שנתגלו במהלך הפרויקט, חומרתם והסבירות שלהם והאמצעים שננקטו כדי לטפל בהם.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| מסמך | סיכון | סבירות | חומרה | פירוט הסיכון | מידת ההשפעה על הפרויקט והתקדמות המערכת | תצורת הטיפול |
| מסמך יזום + מסמך הנדסי | ניתוח דרישות לא מספק | 3 | גבוהה | ניתוח דרישות לא מספק מהווה סיכון משמעותי לפרויקט. הדבר כרוך באי הבנה או תיעוד יסודי של הצרכים והאילוצים של המערכת, מה שמוביל לאי-התאמה פוטנציאלית בין המערכת המסופקת לדרישות בפועל. | השלב הראשוני של הפרויקט היה לעמוד אחר הדרישות הפונקציונליות ולא הפונקציונליות של המערכת על מנת להימנע מסיכון זה, שכן לסיכון זה פוטנציאל עיכוב משמעותי לפרויקט. יצרנו מסמך אפיון מפורט של המערכת כולל תרחישי שימוש בה כדי להימנע מסיכון זה. בנוסף, הגישה האיטרטיבית והתקשורת המשופרת בין חברי הצוות הפחיתו את ההשפעות ארוכות הטווח, ואפשרו לפרויקט להתיישר עם הציפיות שהוגדרו. | אומצה גישת פיתוח איטרטיבית, המאפשרת הערכה שוטפת והתאמה של דרישות לאורך מחזור חיי הפרויקט. הדבר הבטיח שניתן לשלב צרכים מתפתחים בתכנון ובפונקציונליות של המערכת. בנוסף, המערכת נבנתה בצורה מודולרית כך שגם אם היה צורך בשינוי, חלקי הקוד ששונו בפועל היו קטנים (יחסית) ולא גרמו לנו לכתוב מחדש חלקי קוד רבים. |
| שגיאות באלגוריתם | 5 | גבוהה | שגיאות באלגוריתם עלולות לגרום לאי הקפדה על נורמות ריחוק חברתי, מה שיוביל לסיכונים בריאותיים ומשפטיים. בנוסף, שגיאות באלגוריתם משמעותם חוסר עניין במערכת מצד משתמשים שמוביל באופן ישיר לירידה בשביעות רצון המשתמשים. | באגים באלגוריתם היוו סיכון משמעותי למטרות הפרויקט ועלולות לפגוע באמינותו וביעילותו. הטמעת מנגנוני בדיקות ומנגנונים קפדניים דרשו זמן ומשאבים נוספים, מה שהוביל לעיכובים קלים בציר הזמן הכולל של הפרויקט. עם זאת, אמצעים אלה היו חיוניים כדי להבטיח שהמערכת אמינה ובטוחה, ובסופו של דבר שומרים על אמון המשתמש ועמידה בהנחיות הבריאות. (בקוד המערכת יש package המכיל בדיקות לכל תרחיש). | בדיקות קפדניות: נוהלי בדיקה מקיפים יושמו כדי לזהות ולתקן כל שגיאה באלגוריתם. הדבר כלל גם תרחישים מדומים וגם תרחישים בעולם האמיתי כדי להבטיח את חוסנו של האלגוריתם.  כאשר משתמש רוצה להזמין מקומות באולם- הוא מציין את סוג המחלה בה הוא חולה ובהתאם לכך המערכת שומרת מקום בהתאם. במערכת שנבנתה יש 2 אופציות- חולה קורונה/ לא חולה קורונה. במידה והמשתמש רוצה להזמין כרטיסים עבור אנשים חולים, המשתמש מציין את סוג המחלה (המערכת גמישה כדי להתאים את עצמה למחלות נוספות בהמשך). |
| אופטימיזציה של מקומות ישיבה תוך שמירה על ריחוק חברתי עשויה להיות מורכבת מבחינה חישובית ולא תמיד תייצר פתרון אופטימלי בשל האילוצים הכרוכים בכך. | 8 | גבוהה מאוד | אופטימיזציה של סידורי הישיבה תוך שמירה על נורמות ריחוק חברתי היא מורכבת מבחינה חישובית ולא תמיד תייצר פתרון אופטימלי בשל האילוצים הכרוכים בכך. מורכבות זו עלולה להוביל לזמני עיבוד ארוכים יותר ולהקצאות מושבים לא אופטימליות, ולהשפיע על יעילות המערכת ושביעות רצון המשתמש. | המורכבות של ייעול הישיבה תוך שמירה על נורמות הריחוק החברתי הציגה אתגרים משמעותיים. פיתוח ויישום אלגוריתמים וגישות היוריסטיות מתקדמים דרשו מאמצי מחקר ופיתוח משמעותיים. עם זאת, צעדים אלה היו נחוצים כדי להבטיח שהמערכת תוכל לנהל ביעילות את סידורי הישיבה תחת אילוצים משתנים, ובסופו של דבר תבנה מערכת שתוכל לענות על שביעות רצון המשתמשים. הטיפול המוצלח בסיכון זה חיזק את יכולת המערכת לספק פתרונות ישיבה אמינים ואחראיים חברתית, תוך התאמה עם היעדים הכוללים של הפרויקט. | פיתוח והטמעה של אלגוריתמי אופטימיזציה מתקדמים שתוכננו במיוחד להתמודד עם אילוצים מורכבים ביעילות. אלגוריתמים אלו שואפים למצוא את סידורי הישיבה הטובים ביותר במסגרת המגבלות הנתונות.  גישות היוריסטיות: שילוב שיטות היוריסטיות המספקות פתרונות כמעט אופטימליים (ביחס לאילוצים, אי אפשר לספק את כל המשתמשים ב100% ולכן הסתפקנו בקצת פחות מ100%) מהר יותר מאשר טכניקות אופטימיזציה מסורתיות. גישות אלו מאזנות בין איכות הפתרון ליעילות חישובית.  אופטימיזציה מצטברת: הטמעת תהליכי אופטימיזציה מצטברים המתאימים את סידורי הישיבה בצעדים קטנים יותר ניתנים לניהול, הפחתת העומס החישובי ושיפור ההיענות.  פירוט אחר אופן פעולת האלגוריתם תועד בהמשך. |
| אי שימוש במערכת מצד משתמשים | 3 | 4 | שימוש נמוך במערכת, וכתוצאה מכך אי השגת יעדי הפרויקט ובזבוז השקעה. | בסופו של דבר, המערכת נבנתה הן בשביל המשתמשים (המעוניינים לקנות כרטיסים) והן בשביל בעלי האולמות. אי שימוש במערכת מצד הלקוחות, יוביל באופן ישיר לירידה בהכנסות עבור בעלי האולמות, דבר שיבנה מערכת עם אלגוריתם חזק ויעיל אך לא שימושית בפועל. | יצרנו חווית משתמש נעימה ואטרקטיבית (המוצגת בגרסת האלפא של הפרויקט) כדי למשוך את המשתמשים להשתמש במערכת ולאמץ אותה כאשר מעוניינים לרכוש כרטיסים. |

**1.4 אופן ההתמודדות עם המגבלות שהוגדרו במסמכי התכן הקודמים-**

פרק זה מפרט את האסטרטגיות והמתודולוגיות שהופעלו כדי לטפל במגבלות שהוגדרו במסמכי התכנון הקודמים. הוא כולל את הידע ההנדסי שהושלם ונלמד, השיטות שיושמו ופעילויות עם לקוחות חיצוניים ופנימיים כדי להבטיח יישום מוצלח של המערכת.

כוח אדם מוגבל מכיוון שאנחנו השניים היחידים שעובדים על הפרויקט.

תיעדוף משימות: ביצענו תיעדוף משימות לאחר כל שלב בפיתוח הפרויקט, כדי להתמקד בהיבטים הקריטיים ביותר ולהבטיח שאלה ייושמו בתחילה.

זרימות עבודה יעילות: הטמענו מתודולוגיות זריזות כדי לשפר את שיתוף הפעולה והיעילות. אוטומציה: השתמשנו בקוד אוטומציה למשימות חוזרות כמו בדיקות (יש בקוד הפרויקט package המכיל בדיקות אוטומטיות), מה שעזר לנו בחיסכון בזמן ובהפחתת מאמץ ידני.

מגבלות זמן ופיתוח:

פיתוח מערכת מאפס יכול לקחת הרבה זמן. זה עלול להוביל לשלבי פיתוח ממהרים, שעלולים לגרום לבאגים ולבעיות במערכת.

השתמשנו בתהליך פיתוח איטרטיבי כדי לשכלל ולשפר באופן מתמיד את המערכת. זה עוזר בזיהוי ותיקון באגים בתחילת מחזור הפיתוח.

בדיקות מתמשכות: הטמענו בדיקות אוטומציה כדי להבטיח בדיקות תכופות וזיהוי מוקדם של בעיות טרם שחרור הקוד.

תזמון ריאלי: הגדרנו זמנים מציאותיים כדי להימנע משלבי פיתוח ממהרים שאינם עונים לדרישות.

תכנון ספרינט בהתאם להגשת הדוחות, אפשר לנו לשמור על זרימת עבודה דינמית וגמישה. על ידי פירוק הפרויקט לספרינטים ניתנים לניהול, התמקדנו בהתקדמות מצטברת, תוך אספקת רכיבים פונקציונליים של המערכת באופן קבוע.

מורכבות חישובית

אלגוריתמים המחפשים פתרונות אופטימיזציה גלובליים עשויים להפוך לבלתי פתירים מבחינה חישובית עם הגדלת גודל הבעיה (כלומר, מספר גדול יותר של מושבים והעדפות מורכבות).

אלגוריתמים היוריסטיים: השתמשנו באלגוריתמים היוריסטיים ובקירוב המספקים פתרונות כמעט אופטימליים עם מורכבות חישובית מופחתת. בנוסף, בתחילה המערכת תוכננה כך שעבור כל לקוח המעוניין לבצע הזמנה, נמצא פתרון חמדני. לאחר שכלל הלקוחות יקבלו פתרון חמדני המערכת תיסגר להזמנות ולאחר מכן נפעיל אלגוריתם אופטימיזציה גבולית על כל המערכת כדי למצוא את הסידור הטוב ביותר (שידע למקסם את הרווחים ולהתחשב במגבלות הבריאות ואילוצי המשתמשים). לאחר שהבנו שפיתוח מערכת זאת לא ישים הן מבחינת הלקוחות והן מבחינת זמני התגובה (בעיקר עבור אולמות שבהם כמות המושבים גדולה מאוד), שינינו את אפיון המערכת כך שכעת המערכת תאפשר לכל לקוח לבצע הזמנה, המערכת תמצא עבור כל לקוח פתרון חמדני להושבה ולאחר מכן תנסה למצוא פתרון אופטימלי בהתחשב בסך האילוצים. כלומר, המערכת מייצרת עבור כל לקוח פתרון אופטימלי ולא על כל הושבת האולם יחד. הפתרון האופטימלי מבוסס אלגוריתם היוריסטי בעל פונקציה מטרה שמטרתה למקסם את הרווחים עבור בעלי האולמות, תוך התחשבות בנהלי הבריאות העדכניים.

בדיקות התוכנה-

קוד הפרויקט שלנו כולל חבילה המכילה בדיקות. בדיקות אלו עזרו לנו לשמור על שלמות המערכת שלנו על ידי אימות מתמשך של הפונקציונליות שלה לאחר כל שינוי קוד.

**2 מפרטי תכן וארכיטקטורת המערכת שמומשה במסגרת הפרויקט-**

**2.2 תרשים ארכיטקטורה מפורט-**

A diagram of a software system

Description automatically generated

**2.3 התייחסות מפורטת לכל בלוק ומודל-**

Presentation Layer- מכילה את User Interface (UI) Blocks –

Seat Reservation Interface - מספקת את הממשק שבו משתמשים יכולים לבחור מושבים ומועדים לצפייה. כמו כן, למשתמשים קיימת האופציה להזמין מספר כרטיסים, לפצל את הזמנתם, להכניס אופציה האם האנשים חולים/ בריאים (כדי להתחשב באילוצי הקורונה, ומחלות אחרות ואילוצים נוספים של המשתמשים) ועוד. הממשק מאפשר למשתמשים לצפות במושבים פנויים, לבצע בחירות ולאשר הזמנות.

A screenshot of a phone

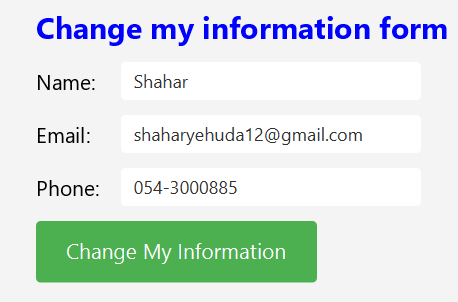
Description automatically generatedA screenshot of a menu

Description automatically generated

מודול זה משפר את חווית המשתמש על ידי מתן ממשק אינטואיטיבי ואינטראקטיבי להזמנת מושבים. זוהי נקודת המגע העיקרית של המשתמשים לאינטראקציה עם המערכת.

User Account Interface- בתפריט המוצג למשתמש, יש אופציה לערוך את פרטיו (השם, המייל והסיסמה) וכל אלה משתנים בDB .

כאשר לוחצים על הכפתור בתפריט המאפשר לשנות את הגדרות המשתמש, נפתחת למשתמש האפשרות לערוך את פרטיו-



Business Logic Layer- מכילה את מודול הזמנת מושבים, מודול חשבון המשתמש ואלגוריתם האופטימיזציה- ליבת המערכת.

Seat Reservation Module- מטפל בבחירת מושב ובקשות להזמנה. השכבה בודקת את זמינות המושבים, מעבדת הזמנות הנקלטות ע"י המשתמש ומבטיחה שמקפידים על הנחיות התרחקות חברתית בהתאם לכללי הבטיחות הבריאותיים.

מודול זה חיוני לפונקציונליות הליבה של המערכת, המבטיח שהזמנות מושבים מטופלות ביעילות ובדייקנות.

User Account Module- מנהלת חשבונות משתמש, מעבדת עדכונים על המשתמש ומטפלת באימות והרשאה.

מודול זה מבטיח ניהול מאובטח ויעיל של נתוני המשתמש, תורם לאבטחת המערכת ולאמון המשתמש.

Optimization Engine- האופטימיזציה היא מרכיב קריטי במערכת הקצאת המושבים. תפקידו העיקרי הוא ליישם אלגוריתמי אופטימיזציה והיוריסטיקה עבור כל הזמנה כדי להבטיח את הקצאת המושבים היעילה ביותר תוך מיקסום ההכנסות ושמירה על הנחיות התרחקות חברתית ואילוצי המשתמש. האלגוריתם בכללותו ממומש באופן הבא-

כשאדם מגיע ורוצה לבצע הזמנה, הפונקציה tryToAllocateInvitation (הנמצאת במחלקת Event בקוד) מתחילה לרוץ.

בהתחלה הפונקציה מנסה למצוא מקומות ביחד לפי המחיר המבוקש

אם הפונקציה לא מצאה, היא מנסה למצוא מקומות מופרדים במחיר המבוקש

במידה ולא מצליחה למצוא מקומות בנפרד, הפונקציה מנסה למצוא מקומות ביחד במחיר אחר

במידה והפונקציה לא מצליחה למצוא הושבה, מנסה למצוא מקומות בנפרד במחיר אחר

אם לא, המערכת מודיעה למשתמש כי לא ניתן להשלים את ההזמנה בהתאם לאילוצים.

אלגוריתם האופטימיזציה (ליבת המערכת) מאזן בין מספר אילוצים והעדפות משתמש כדי לספק סידורי ישיבה אופטימליים תוך שימוש בהיוריסטיקה שפועלת באופן הבא-

פונקציית המטרה לוקחת את ההפסד מההושבה שהתקבלה מהפתרון החמדני ומכפילה ב10, לאחר מכן, הפונקציה מורידה את הכמות המינימלית של המקומות שנשארו ליד וכופלת ב5 ולבסוף מורידה את כמות החורים שנסתמו.

פונקציית המטרה כוללת 3 משתנים- הפסד (נמדד בכסף) מסביב להושבה שהתקבלה במקום מסוים, כמות מינימלית להזמנה של מקומות ליד (כלומר, כמה מקומות פנויים נשארו ליד ההושבה שהתקבלה) וכמות חורים שההזמנה סותמת.

נרצה למקסם את הרווחים עבור בעלי האולמות ולכן ניתן משקל יותר גדול להפסדים (מוכפל ב10) ולאחר מכן המטרה היא להשאיר מקום מקסימלי ביותר ליד או 0 (לכן ההכפלה ב5) כדי למזער את מספר המושבים הריקים הסמוכים ולבסוף המטרה היא לסתום כמה שיותר חורים (לא מוכפל בדבר).

ההיוריסטיקה באלגוריתם היא היוריסטיקה לוקלית מאחר ומבצעת חיפוש היוריסטי מסביב לפתרון החמדני שהתקבל בפתרון ההושבה הראשוני. הפונקציה לא מחפשת הושבה אופטימלית על כל האולם אלא רק על מושבים שנמצאים 10 מקומות מסביב לפתרון החמדני. מצאנו שזאת הדרך היעילה ביותר לחפש הושבה אופטימלית, מאחר וחיפוש על כל האולם הוא לא ישים ולא יעיל (בעיקר עבור אולמות גדולים בהם מספר המושבים באולם גדול מאוד). הפונקציה מבוססת משקולות ומאחר ומטרת העל של הפרויקט היא להוביל למיקסום הרווחים עבור בעלי האולמות, המשקל הגבוה ניתן להפסד (הנמדד בכסף) שבעלי האולמות צפויים להפסיד. הפונקציה מחשבת עבור כל הושבה אופציונלית מסביב להושבה שהתקבלה בפתרון החמדני (עד מרחק 10) את הערך ולבסוף לוקחת את המינימלי מבניהם.

המטרה הנוספת היא להשאיר כמה שיותר מקומות פנויים ליד ההושבה (או אפס) כדי לאפשר לבצע בהמשך הזמנה יותר גדולה (המקרה האידיאלי הוא שמסביב להושבה שהתקבלה יישארו לנו 0 מקומות פנויים, במקרה זה פונקציית המטרה תשתמש במספר המקסימלי מבין כמות השורות / עמודות כדי להוריד כמה שיותר מפונקציית המטרה).

המטרה הנוספת היא לסתום כמה שיותר חורים ולכן מתחשבים גם בכמות סתימת החורים בחישוב הערך של פונקציית המטרה.

תיאור מתמטי של פונקציית המטרה-

L - ההפסד בהכנסות מסידור הישיבה הנוכחי.

P - מספר המושבים הפנויים שנותרו ליד סידור הישיבה הנוכחי.

H- להיות מספר ה"חורים" שלא מולאו בסידור הישיבה.

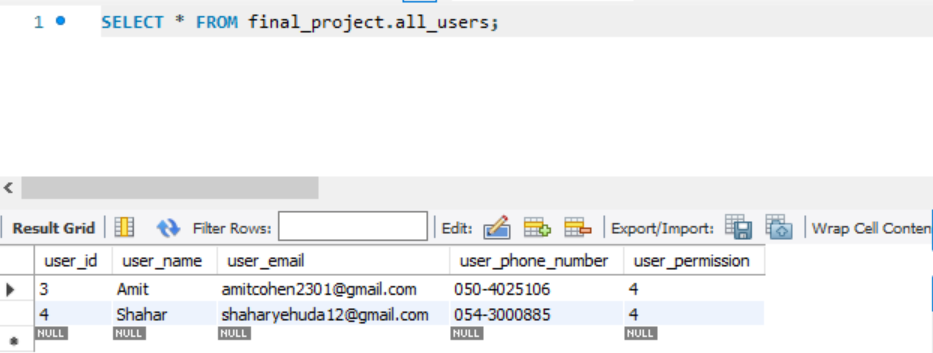
המערכת מעריכה את כל סידורי הישיבה האפשריים בטווח מוגדר (במערכת שלנו, עד מרחק של 10) ובוחרת את זה שממזער את F, מה שמוביל לסידור הישיבה האופטימלי בהתבסס על האילוצים והעדיפויות הנתונות.

Data Layer-

Database Server (MySQL)- מאחסן את כל הנתונים הקבועים כגון פרופילים של משתמש, רשומות הזמנות ופרטי צפייה. מבטיח שלמות הנתונים ומטפל באחזור נתונים ועדכונים.

מודול זה מספק פתרון אחסון נתונים חזק ואמין, המבטיח שכל נתוני המערכת מאוחסנים בצורה מאובטחת ונגישים בקלות לעיבוד.

דוגמה לשמירת המשתמשים שלנו בDB –



שמירת הרשאות משתמשים בDB –

A screenshot of a computer

Description automatically generated

דוגמה לשמירת אירועים בDB (במקרה הנ"ל נשמר רק אירוע אחד)-

A screenshot of a computer

Description automatically generated

Data Access Objects (DAOs)- משמש כמקשר התקשורת בין שכבת הלוגיקה העסקית למסד הנתונים של MySQL. מבצע פעולות CRUD (צור, קריאה, עדכן, מחק) במסד הנתונים.

מודול זה מקל על ניהול נתונים יעיל על ידי מתן דרך לאינטראקציה עם מסד הנתונים, תוך הבטחת זרימת נתונים חלקה בין האפליקציה לאחסון הנתונים.

זרימת המידע-

Seat Reservation Interface ↔ Seat Reservation Module- - מציין שממשק הזמנת המושב שולח בקשות להזמנה למודול הזמנת המושב ומקבל תגובות.

User Account Interface ↔ User Account Module- - מציין שממשק חשבון משתמש מתקשר עם מודול חשבון משתמש לניהול חשבונות משתמש.

Seat Reservation Module ↔ DAO- - מציין שמודול חשבון משתמש מקיים אינטראקציה עם ה-DAO כדי לנהל את נתוני חשבון המשתמש.

Optimization Engine ↔ DAO- - מציין שמנוע האופטימיזציה מקיים אינטראקציה עם ה-DAO- כדי לקבל נתונים הנדרשים למיטוב ולעדכון הקצאת מושבים.

DAO ↔ Database- - מציין שה-DAO מבצע שאילתות מסד נתונים כדי ליצור אינטראקציה עם שרת מסד הנתונים.

**2.4 מפרטי אלגוריתמים-**

סעיף זה מספק מפרטים מפורטים של האלגוריתם במערכת הקצאת המושבים בבית קולנוע. הוא כולל מידע על שפות התכנות, תצורת אינטגרציה של מודול, קשרי עיצוב וארכיטקטורת מבנה, ודיאגרמת קשרי הישות (ERD). בנוסף, הוא מכסה את הממשקים והקישוריות בתוך המערכת.

שפות תכנות וסביבת פיתוח

שפת תכנות: Java

סביבת פיתוח: Eclipse IDE 2022-12

בסיס נתונים: MySQL

Module Integration Configuration-

Seat Reservation Interface –

מאפשרת למשתמשים לבחור מושבים ומועדי אירועים לצפייה.

מתקשר עם מודול שמירת המושב בשכבת ההיגיון העסקי.

User Account Interface-

ניהול חשבונות משתמש ופרטי הזמנות עבור כל הזמנה של המשתמש.

מקיים אינטראקציה עם מודול חשבון המשתמש בשכבת הלוגיקה העסקית.

Seat Reservation Module-

מטפל בבחירת מושב ובקשות להזמנה.

מקיים אינטראקציה עם ממשק הזמנת מושבים ומודול ה-DAO כדי לבדוק זמינות מושבים ולעבד הזמנות.

User Account Module-

ניהול פרטי חשבון משתמש והזמנות.

מתקשר עם ממשק חשבון המשתמש ומודול ה-DAO לניהול נתוני משתמש.

Optimization Engine-

מיישם אלגוריתם אופטימיזציה עבור הקצאת המושבים בהתאם לאילוצי המשתמש ותקנות הבריאות.

מקיים אינטראקציה עם מודול ה-DAO כדי להביא נתונים ולעדכן הקצאת מושבים.

Data Access Objects (DAOs)-

מקל על תקשורת בין שכבת הלוגיקה העסקית למסד הנתונים של MySQL.

אינטגרציה: מבצעת פעולות CRUD על מסד הנתונים.

Database Server (MySQL) –

מאחסן את כל הנתונים הקבועים כגון פרופילים של משתמש, רשומות הזמנות ופרטי צפייה.

מקיים אינטראקציה עם מודול ה-DAO כדי לאחזר ולעדכן נתונים.

Design Relationships and Structure Architecture-

ארכיטקטורת המערכת בנויה סביב מודל שלוש שכבות: שכבת המצגת, שכבת ההיגיון העסקי ושכבת הנתונים. ארכיטקטורת שכבות זו הופכת את המערכת לניתנת לניהול, להרחבה ובטוחה יותר.

Presentation Layer - שפת תכנות: Java ו-JavaFX עבור העיצוב תוך שימוש בCSS. שכבה זו אחראית לאינטראקציה של המשתמש. היא מציגה את ממשק המשתמש ומעבירה את פעולות המשתמש לשכבת הלוגיקה העסקית. רכיבי השכבה מורכבים ממשק משתמש שהם מסכים המוצגים למשתמש באמצעות תפריט. דרך המסכים המשתמשים מקיימים אינטראקציה עם המערכת.

-Business Logic Layer שפת תכנות: Java .שכבה זו מעבדת נתונים משכבת ממשק המשתמש (UI), מיישמת כללים עסקיים (לוגיקה של הזמנת מקומות) ומקיימת אינטראקציה עם שכבת הנתונים.

רכיבי השכבה מורכבים ממודול הזמנת מושבים שמטפל בכל הפונקציות הקשורות לבחירת מושבים והזמנת מושבים וממנוע אופטימיזציה שמיישם את האלגוריתמים הגלובליים לאופטימיזציה והיוריסטיים.

שכבת נתונים- טכנולוגיה: MySQL Database. השכבה אחראית על אחסון, אחזור וניהול נתונים. היא מאחסנת את נתוני משתמש, הצגת פרטים, ומכילה מידע על מושבים ורשומות הזמנות. רכיבי השכבה הם שרת מסד נתונים שמארח את מסד הנתונים של MySQL. מחלקות Java המתקשרות עם מסד הנתונים, מבצעות שאילתות ומעדכנות נתונים.

אופן פעולת האלגוריתם-

האלגוריתם מתחיל בגישה חמדנית להקצאת מושבים עבור כל הזמנה הנכנסת למערכת. שלב ראשוני זה חיוני כדי למצוא במהירות פתרון בר-קיימא ללא חישוב מקיף. פעולת הפתרון החמדני:

בדיקת מושבים ראשונית: האלגוריתם מתחיל בסריקת פריסת הישיבה כדי למצוא בלוק של מושבים עוקבים (במידה המושבים צריכים להיות רציפים) שיכולים להכיל את כל הקבוצה ברמת המחיר שצוינה.

ניתנת עדיפות למציאת מושבים רצופים במחיר המבוקש תחילה.

אם פנויים X מושבים רצופים בY מחיר כל אחד, הם נשמרים מיד להזמנה. הפתרון החמדני מספק את בקשת המשתמש עם חישוב מינימלי.

אם הפתרון החמדני לא מצליח למצוא בלוק של X מושבים רצופים במחיר המבוקש, הוא ממשיך לבדוק תצורות אחרות (פיצול ההזמנה).

אופן פעולת האלגוריתם באופן כללי-

בהתחלה הפונקציה מנסה למצוא מקומות ביחד לפי המחיר המבוקש

אם לא הפונקציה לא מצאה, היא מנסה למצוא מקומות מופרדים במחיר המבוקש

במידה ולא מצליחה למצוא מקומות בנפרד, הפונקציה מנסה למצוא מקומות ביחד במחיר אחר

במידה והפונקציה לא מצליחה למצוא הושבה, מנסה למצוא מקומות בנפרד במחיר אחר

אם לא, המערכת מודיעה למשתמש כי לא ניתן להשלים את ההזמנה בהתאם לאילוצים.

לאחר שהמערכת מוצאת פתרון הושבה באופן חמדני, המערכת עוברת לשלב האופטימיזציה המבוסס על אופטימיזציה היוריסטית לוקלית-

אלגוריתם האופטימיזציה (ליבת המערכת) מאזן בין מספר אילוצים והעדפות משתמש כדי לספק סידורי ישיבה אופטימליים תוך שימוש בהיוריסטיקה שפועלת באופן הבא- פונקציית המטרה עליה מבוססת ההיוריסטיקה לוקחת את ההפסד מההושבה שהתקבלה מהפתרון החמדני ומכפילה ב10, לאחר מכן, הפונקציה מורידה את הכמות המינימלית של המקומות שנשארו ליד וכופלת ב5 ולבסוף, מורידה את כמות החורים שנסתמו.

פונקציית המטרה כוללת 3 משתנים- הפסד (נמדד בכסף) מסביב להושבה שהתקבלה במקום מסוים, כמות מינימלית להזמנה של מקומות ליד (כלומר, כמה מקומות פנויים נשארו ליד ההושבה שהתקבלה) וכמות חורים שהזמנה סותמת.

נרצה למקסם את הרווחים עבור בעלי האולמות ולכן ניתן משקל יותר גדול להפסדים (מוכפל ב10) ולאחר מכן המטרה היא להשאיר מקום מקסימלי ביותר ליד או 0 (לכן ההכפלה ב5), כדי למזער את מספר המושבים הריקים הסמוכים.

ההיוריסטיקה באלגוריתם היא היוריסטיקה לוקלית מאחר ומבצעת חיפוש היוריסטי מסביב לפתרון החמדני שהתקבל בפתרון ההושבה הראשוני. הפונקציה לא מחפשת הושבה אופטימלית על כל האולם (אצלנו במערכת, הפונקציה מחפשת 10 מושבים מסביב לפתרון החמדני). מצאנו שזאת הדרך היעילה ביותר לחפש הושבה אופטימלית, מאחר וחיפוש על כל האולם הוא לא ישים ולא יעיל (בעיקר עבור אולמות גדולים בהם מספר המושבים באולם גדול מאוד). הפונקציה מבוססת משקולות ומאחר ומטרת על של הפרויקט היא להוביל למיקסום הרווחים עבור בעלי האולמות, המשקל הגבוה ניתן להפסד (הנמדד בכסף) שבעלי האולמות צפויים להפסיד. הפונקציה מחשבת עבור כל הושבה אופציונלית מסביב להושבה שהתקבלה בפתרון החמדני (עד מרחק 10) את הערך ולבסוף לוקחת את המינימלי מבניהם. נדגיש כי בחיפוש ההיוריסטי אין מעבר על כל האולם.

במידה ופונקציית המטרה קיבלה עבור מספר סידורים את אותו הערך, היא תיקח את הסידור הראשון שהתקבל (דוגמה בהרצה שמסופקת בהמשך) והיא תעשה זאת באופן הבא- המערכת תנסה למצוא סידור ישיבה -> תקבל את הערך שהתקבל בפונקציית המטרה -> המערכת תשמור את הערך -> המערכת תשווה בין הערך השמור לערך הבא שהתקבל עבור סידור נוסף -> המערכת תשמור את המינימלי מבין שניהם -> המערכת תעבור לסידור נוסף ותשווה שוב.... (לולאה שרצה כמספר האפשרויות להושבה).

בקוד המשתנים שמתחזקים את הערך של פונקציית המטרה הינם-

numOfLoses - כמות הכסף שמפסידים שבהושבה במקום מסוים (כל המקומות שצריך לשים בהם רווח ומחשב את העלות שלהם).

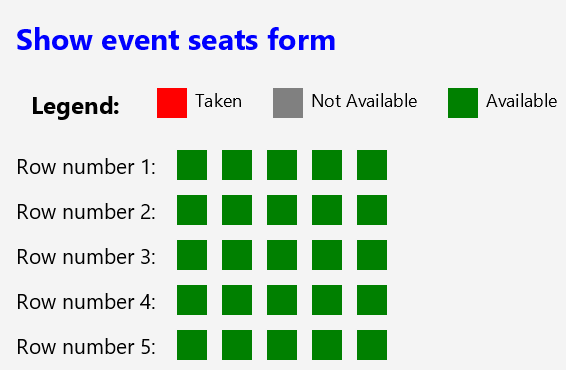
minPlaceToOrder – מספר מקומות פנויים שיישארו ליד, במידה ונחליט להושיב במקום שנמצא.

numOfPlaces - כמה חורים סתמנו (אם נושיב במקום הספציפי אז יהיו לנו X מקומות מסביב שהם 0 כלומר אי אפשר להושיב שם- המצב האידיאלי).

**נראה דוגמת הרצה בקוד-**

נדגיש כי למערכת 2 משתמשים עיקריים- המשתמש המעוניין לבצע הזמנה (לרכוש כרטיסים) ובעלי האולמות אשר מגדירים אירועים לצפייה (סרטים, במקרה שלנו) ופריסות ישיבה של אולמות.

נניח שנרצה לבצע הזמנה עבור האולם שכך נראית פריסת הישיבה בו (האולם ריק מאחר וכל הכיסאות מסומנים כזמינים, צבע ירוק)-



ננסה לקנות 5 כרטיסים במחיר 1$ (כל האולם מתומחר באותו המחיר והוא דולר בודד)-

A screenshot of a invitation

Description automatically generated

המערכת תציג למשתמש את האופציה שמצאה ולמשתמש תינתן האופציה האם לקבל את ההזמנה או לסרב לקבלה (במקרה שלנו, בחרנו שלא לקבל את ההזמנה)-

A screenshot of a invitation

Description automatically generated

נציג את הconsloe –

A screenshot of a computer program

Description automatically generated

הערך המינימלי שהתקבל לפונקציית המטרה עבור כל ההרצות הוא 20 (ניתן לראות זאת בשורה המודגשת) ולכן האלגוריתם בחר בהושבה בשורה 1 (כל השורה כי המשתמש בחר 5 מושבים ושורה 1 מכילה 5 מושבים ריקים).

נראה דוגמת הרצה בקוד, כאשר המשתמש מנסה לקבל מושבים במחיר גבוה (כל מושב) ואין באולם מושבים במחיר זה (יש במחירים נמוכים יותר)-

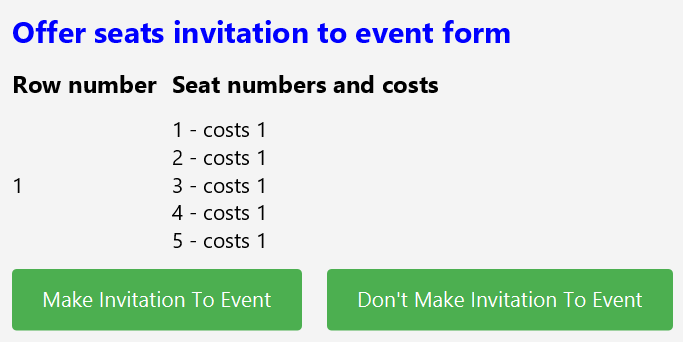
פריסת ההושבה נותרת אותו הדבר (מאחר ובחרנו שלא לקבל את ההזמנה הקודמת)

ננסה לקנות 5 כרטיסים במחיר 5$ כ"א (נציין שוב כי אין באולם כרטיסים במחיר המבוקש)

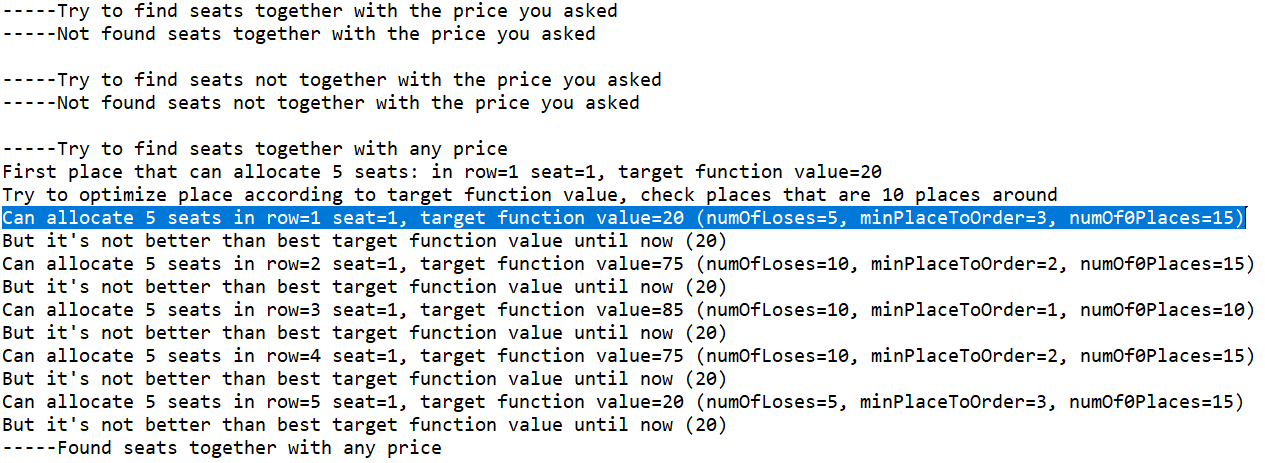
A screenshot of a invitation

Description automatically generated

המערכת תציע למשתמש את השורה הראשונה-



נציג את הconsloe –



המערכת ניסתה להושיב יחד במחיר המבוקש ולא מצאה -> המערכת ניסתה להושיב יחד במחיר אחר מהמבוקש ולא מצאה -> המערכת ניסתה להושיב יחד במחיר אחר מהמבוקש והצליחה מאחר וכל הכיסאות באולם מתומחרות במחיר 1$. ניתן לראות גם כאן שהערך המינימלי של פונקציית המטרה הינו 20 והוא התקבל עבור ההושבה בשורה הראשונה ובשורה החמישית. מאחר והאלגוריתם בוחר את הסידור הראשון שמצא במידה והתקבלו 2 ערכים זהים לפונקציית המטרה, נבחרה שורה ראשונה.

**ERD-**

A screenshot of a computer diagram

Description automatically generated

**2.5 פירוט תצורות ותכולות מעטפת של התוצרים, המודולים והמערכת-**

סעיף זה מספק מבט מעמיק על התצורות ותכולת המעטפת של המוצרים, המודולים והמערכת הכוללת במסגרת הפרויקט. זה גם מדגיש את הזרימה והפעולות של האלגוריתם, תוך פירוט תהליך שלב אחר שלב וטכניקות אופטימיזציה שלו.

סקירת מערכת

המערכת פותחה באמצעות Java בתוך ה-Eclipse IDE, עם MySQL כבסיס הנתונים. המערכת בנויה בשלוש שכבות עיקריות: שכבת מצגת, שכבת לוגיקה עסקית ושכבת נתונים. הארכיטקטורה נועדה להבטיח מודולריות, מדרגיות וקלות תחזוקה.

זרימת הנתונים והאינטראקציות בין רכיבים אלה מומחשת בתרשים הארכיטקטורה. לכל מודול יש תפקיד ספציפי, התורם לפונקציונליות הכללית ולחוסנה של המערכת.

זרימה מפורטת של האלגוריתם-

הליבה של המערכת היא אלגוריתם הקצאת המושבים שלה, שמטרתו למקסם את הרווחים עבור בעלי האולמות, תוך הקפדה על העדפות המשתמשים והנחיות ההתרחקות החברתיות. האלגוריתם מבצע את השלבים הבאים:

האלגוריתם מחפש תחילה פתרון ראשוני חמדני, שהוא פתרון מהיר המבוסס על זמינות ורווחיות מיידיות. הפתרון החמדני מניב הושבה אופציונלית ללקוח, שלא בהכרח תהיה ההושבה הסופית. לאחר שמתקבל פתרון חמדני, המערכת עוברת לשלב האופטימיזציה בה היא תנסה למצוא סידור הושבה יעיל ככל האפשר (מבחינת רווחים ומבחינת הושבה בהתבסס על אילוצי המשתמש ואילוצי הקורונה).

עבור המחלה COVID-19, האלגוריתם אוכף מקום של 5 מושבים בכל צד של מושב מוקצה, מה שמבטיח עמידה בהנחיות הבריאות. עבור אנשים בריאים, האלגוריתם אוכף מקום אחד בודד מכל הזמנה הושבה. נציין בנוסף כי המערכת נבנתה בצורה ניתנת להרחבה, כך שאם נרצה להרחיב את האופציות למחלות נוספות בהמשך, נוכל לעשות זאת בקלות.

פתרון חמדני-

עבור משתמשים המעוניינים להזמין מושבים רציפים (+מחיר שהמשתמש מקליד)- האלגוריתם מנסה למצוא בלוק של מושבים רציפים התואמים את המחיר המבוקש.

אם האלגוריתם מצליח, הוא ממשיך להקצות מושבים ללקוח.

מושב בנפרד במחיר המבוקש- אם בלוק רציף אינו זמין, האלגוריתם מציע למשתמש מושבים בודדים (מפוצלים) במחיר המבוקש שיכולים להכיל את כל הקבוצה ומציג אותם למשתמש. למשתמש ניתנת האופציה לקבל את ההצעה או לסרב.

הושבה של כולם ביחד בכל מחיר- אם המחיר המבוקש אינו בר ביצוע, האלגוריתם מרחיב את החיפוש שלו לכל מחיר זמין, תוך עדיפות להושיב את כולם יחד.

מושב בנפרד בכל מחיר- כמוצא אחרון, האלגוריתם מחפש מושבים בודדים בכל מחיר, ומבטיח שכל הלקוחות יושבים, גם אם לא ביחד.

קריטריוני אופטימיזציה- האלגוריתם מעריך כל סידור ישיבה פוטנציאלי על סמך שני קריטריונים עיקריים:

ייעול רווחים- כאשר שמירה על רווחים היא בראש סדר העדיפויות, האלגוריתם בוחר מושבים שממקסמים את ההכנסה, לכן, פונקציית המטרה בעלת המשקל הגבוה ביותר לקריטריון זה (משקל 10 לעומת משקלים 5 ו1 עבור שאר הקריטריונים).

מזעור רווחים- עבור הזמנות ללא דרישת מרווח (לקוחות בריאים), האלגוריתם שואף להשאיר מרווח רב ככל האפשר סביב המושבים שנבחרו, או ללא מרווח כלל, אם זה אופטימלי.

אופן פעולת האלגוריתם הוסברה בפירוט בפרקים הקודמים.

מטרות, יעדים ומדדים-

מטרות-

מתן אפשרות ללקוחות לבצע הזמנה להקרנה ספציפית (אירוע)- מומש באופן מלא.

מתן אפשרות ליצור ולמחוק אירוע -מומש באופן מלא.

מתן אפשרות לערוך אירוע -מומש באופן מלא.

מתן אפשרות להציג בפני המשתמש את האירועים הזמינים להזמנה -מומש באופן מלא.

מתן אפשרות לשנות את ההרשאות של משתמש -מומש באופן מלא.

יעדים-

התממשקות המערכת עם הDB-מומש באופן מלא.

מקסום הכנסות עבור בעלי האולם -מומש באופן מלא.

מימוש אלגוריתם אופטימיזציה אופטימלי לפתרון בעיית הקצאת המושבים -מומש באופן מלא.

הטמעת ממשק אינטראקטיבי, חזותי וידידותי למשתמש – נעשה ע"י שימוש בJavafx- -מומש באופן מלא.

עמידה בתקנות הבריאות כדי לשמור על בריאות הלקוחות -מומש באופן מלא.

מדדים-

ביצענו בדיקות אינטגרציה של המערכת עבור כל אחת מתכונות המערכת לDB.

ביצענו בדיקות אוטומציה ובדיקות ידניות של המערכת עבור כל אחת מתכונות המערכת.

שירותי מערכת:

המערכת מפותחת ב-Java באמצעות Eclipse IDE ומתממשקת עם מסד נתונים MySQL. המשתמש מקיים אינטראקציה עם המערכת באמצעות ממשק משתמש מבוסס JavaFX. להלן, שירותי הליבה של המערכת, הפונקציונליות שלהם והאינטראקציות שלהם מפורטים בהרחבה.

ייעול הקצאת מושבים זהו בעצם ליבת המערכת- שירות זה אחראי על קביעת סידורי הישיבה האופטימליים לכל הזמנה. האלגוריתם מכיל 2 שלבים עיקריים:

אלגוריתם חמדן: מספק פתרון ראשוני להזמנה של המשתמש.

אופטימיזציה מבוססת היוריסטיות: מחדד את הפתרון הראשוני לשיפור הקצאת המושבים וההכנסה הכוללת.

מבטיח עמידה בהנחיות ההתרחקות החברתית על ידי שמירה על מרווח של 5 מושבים משני צידיו של מושב תפוס עבור מושבים בהם צפויים לשבת אנשים החולים בcovid-19 ומרחק של מושב אחד מאנשים בריאים (כפי שהיה במגפת הקורונה).

האלגוריתם מטפל בבקשות משתמשים מרובות, יושבים יחד במידת האפשר (במידה ואכן רוצים לשבת יחד), ומייעל לרווח מקסימלי (מבחינת כסף) ומינימום שטח מבוזבז.

שירותים מקוונים (on line):

ממשק בחירת מושבים: לאחר שהאלגוריתם מצא סידור ישיבה אופטימלי (או אופציונלי במידה והמערכת לא מצאה סידור אופטימלי), הסידור יוצג למשתמש ולמשתמש תינתן בחירה להמשיך לאישור ההזמנה או לחילופין, לבקש מהמערכת שתמשיך ותחפש עבורו סידור טוב יותר. המערכת מעדכנת את הזמינות באופן מיידי בDB עם ביצוע ההזמנות.

משתמשים יכולים להקליד למערכת את אילוציהם- בחירת האירוע אליו ירצו להזמין כרטיסים, התאריך והשעה, כמות המושבים הרצויה להזמנה, האם האנשים המשתתפים בהזמנה בריאים/חולים (במערכת שלנו, יש אופציה להקליד רק את מחלת הcovid-19 אך המערכת נבנתה בצורה הניתנת להרחבה בקלות עבור מחלות אחרות). כמו כן, יוצג למשתמשים התמחור עבור כל כיסא באולם ע"י מעבר עם העכבר על תצורת האולם (לכל כיסא יש מחיר שונה).

שירותים בזמן אמת (real time):

עדכוני זמינות מושבים בזמן אמת: המערכת מספקת עדכונים בזמן אמת על זמינות המושבים כדי למנוע הזמנות כפולות. במידה והמערכת לא מוצאת מושבים זמינים (עם שכלול העדפות המשתמש + הנחיות הבריאות) היא מתריעה למשתמש על סידור אופציונלי שמצאה (גם הסידור אינו עומד על ציפיותיו באופן מלא) ולמשתמש ניתנת האופציה האם לקבל את הסידור או לסרב לקבלו. אופן פעולה זה נוצר מתוך הידיעה כי לא ניתן יהיה לספק את כל המשתמשים (בכפוף לאילוציהם ולאילוצי ההתרחקות החברתית) ולכן אפשרי וניתן לספקם באופן חלקי, וזה עדיף ולא לספק אותם כלל.

הזמנה בזמן אמת + התראה מיידית על אישור הזמנה: המערכת מעבדת הזמנות ומספקת משוב מיידי למשתמש (המשוב הוא למעשה אישור ההזמנה/אי אישורה).

תאימות לבריאות ובטיחות בזמן אמת- המערכת מספקת סידורי הושבה כך שיתאימו להנחיות הבריאות העדכניות, תוך התאמה בזמן אמת ככל שההנחיות משתנות ובכך שומרת על סביבה בטוחה למשתתפים, חיונית לבריאות הציבור ולעמידה ברגולציה.

תיחום זה ותיאור השירותים מספקים סקירה ברורה של מה שמציעה מערכת הזמנת המושבים האולם הקולנוע, ההיקף התפעולי שלה, והשירותים שהיא נותנת בהקשרים מקוונים ובזמן אמת כאחד. מסגרת זו מבטיחה שלמשתמשים ומנהלי מערכת יש הבנה מקיפה של יכולות ומגבלות המערכת.

מפרטי ממשקים-

להלן מפרטי הממשק המפורטים עבור Java, JavaFX, MySQL וממשקים רלוונטיים אחרים.

מפרטי ממשק Java-

Java היא שפת התכנות המשמשת לפיתוח המערכת.

שכבת הלוגיקה העסקית: מיישמת את פונקציונליות הליבה, לרבות אלגוריתם הקצאת מושבים, ניהול משתמשים, ניהול אירועים.

שיעורי שירות: עטפו את ההיגיון העסקי וספקו שיטות לאינטראקציה עם חלקים שונים של המערכת.

שכבת גישה לנתונים: מתממשקת עם מסד הנתונים לביצוע פעולות CRUD.

מפרטי ממשק JavaFX-

JavaFX משמש לבניית ממשק המשתמש הגרפי (GUI) עבור המערכת. הוא מספק סט עשיר של רכיבי ממשק משתמש ליצירת ממשק ידידותי ונוח למשתמש.

קבצי CSS: סגנון רכיבי ממשק המשתמש לחוויה חזותית טובה יותר.

JavaFX SDK: מספק את הספריות והכלים הדרושים לבניית יישומי JavaFX.

Scene Builder: כלי עיצוב חזותי.

מפרטי ממשק MySQL

תיאור: MySQL היא מערכת ניהול מסד הנתונים המשמשת לאחסון וניהול כל הנתונים הקבועים עבור המערכת. הוא מתממשק עם אפליקציית Java באמצעות JDBC (קישוריות ל-Java Database).

סכמת מסד נתונים: מגדירה את מבנה מסד הנתונים, כולל טבלאות, עמודות וקשרים.

JDBC: מספק ממשק סטנדרטי לחיבור ואינטראקציה עם מסד הנתונים של MySQL.

ספריות מפתח ומסגרות:

MySQL Connector/J: מנהל התקן JDBC המאפשר ליישומי Java ליצור אינטראקציה עם מסדי נתונים של MySQL.

הטבלאות השמורות בDB –

A screenshot of a computer

Description automatically generated

UX,UI,GUI תיאור, צירוף והדגמות-

בתחילת הרצת התוכנית-

A screenshot of a login menu

Description automatically generated

לחיצה על login – (המשתמשת כבר נמצאת בבסיס הנתונים)

A screenshot of a login form

Description automatically generated

לאחר אימות המייל המשתמשת מנווטת למסך הראשי של התוכנית בו מוצג התפריט (המשתמשת רשומה במאגר הנתונים)-

A screenshot of a computer

Description automatically generated

לחיצה על כפתור Create Event (הקרנת סרט, לדוג)-

A screenshot of a computer

Description automatically generated

לאחר מכן נווטים אל מסך בו יש לקבוע את מחירי הכיסאות באולם בהתאם לכמות השורות שהגדרנו ומספר הכיסאות בכל שורה. (נציין כי מחיר 0 משמעותו כי לא ניתן להושיב במקום זה אדם, מה שאומר שבאופן זה אנחנו יכולים לטפל בתצורות אולם שונות, בהמשך)-

A screenshot of a computer

Description automatically generated

בהנחה ותהליך מסתיים בהצלחה, מוצג למשתמש/ת המסך-

A screenshot of a computer

Description automatically generated

לחיצה על הכפתור- Show events seats תציג את המופעים הזמינים לצפייה ואופציה להסתכל על תצורת הישיבה של כל אולם-

A screenshot of a computer

Description automatically generated

ניתן לראות שעבור האירוע הממוספר 0, שורה 1 לא זמינה להזמנה מכוון שהקלדנו את מחירי הכיסאות בשורה זו ב0 שקלים.

A screenshot of a form

Description automatically generated

לחיצה על הכפתור Make Invitation תציג למסך אופציה לבחור אירוע לצפייה (נניח ובחרנו את הסרט Harry Poter -

A screenshot of a computer

Description automatically generated

הזמנה של 2 מקומות עם מחיר של 20$-

A screenshot of a invitation

Description automatically generated

המערכת מוצאת סידור ישיבה שעונה לסט האילוצים ומציגה את הסידור למשתמש, המשתמש בוחר האם לקבל את הסידור או לסרב לקבלו-

A screenshot of a website

Description automatically generated

**תסריטים המבוססים על תרחישי השימוש עיקריים של המערכת-**

הרעיון הכללי בכל התסריטים-

שלבי האלגוריתם:

קודם מחפש מושבים ביחד

אם לא מוצא -מחפש מושבים בנפרד

אם לא מוצא -מחפש מושבים ביחד במחיר משתנה

אם לא מוצא מחפש מושבים בנפרד במחיר שונה

אם לא מוצא מודיע שאין אפשרות

שלב 1- פתרון חמדני ראשוני לבעיית ההושבה. מציאת סידור ישיבה אפשרי, לא בהכרח אופטימלי.

שלב 2- שלב האופטימיזציה- ליבת האופטימיזציה היא פונקציית מטרה המאזנת בין מספר גורמים: אובדן ההכנסות מסידור הישיבה המוצע, מספר המושבים הריקים הסמוכים ומספר ה"חורים" הלא מלאים או המושבים הריקים המבודדים. הפונקציה נותנת את המשקל הגבוה ביותר למזעור אובדן הכנסות, ולאחר מכן מזעור מושבים ריקים סמוכים, ולבסוף מילוי חורים רבים ככל האפשר. על ידי הערכה והשוואה של סידורי הישיבה הפוטנציאליים בהתבסס על קריטריונים אלו, המערכת בוחרת את זה שמציע את האיזון האופטימלי, תוך הבטחת הקצאת מושבים יעילה ורווחיות מרבית.

**תסריט 1- "המקרה הטוב"- האולם ריק ומגיעה משפחה המעוניינת לקנות 5 מושבים רציפים ב-$5 (לדוג') כל אחד-**

פעולת משתמש: המשתמש (נציג המשפחה) מקיים אינטראקציה עם ממשק המשתמש מבוסס JavaFX כדי לבקש 5 מושבים ב-$5 כל אחד.

המערכת בודקת את זמינות הישיבה באולם.

היא מקצה 5 מושבים צמודים אם זמינים, במחיר המבוקש.

המערכת מעדכנת את מצב המושב לשמור עבור המשתמש.

המערכת שומרת עבור הקבוצה 5 מושבים ב-$5 כל אחד, והמושבים מסומנים כתפוסים במערכת.

מבחינת האלגוריתם-

האולם ריק בהתחלה ולכן, כל המושבים פנויים.

מגיעה משפחה (קבוצה של אנשים) ומבקשת לרכוש 5 מושבים רצופים ב-$5 כל אחד.

הפונקציה tryToAllocateInvitation מופעלת עם הבקשה ל-5 מושבים רצופים ב-$5 כל אחד.

חיפוש מושבים ביחד במחיר המבוקש: האלגוריתם מנסה בהתחלה למצוא 5 מושבים רצופים במחיר המבוקש של $5 למושב. מכיוון שהאולם ריק ובאולם יש 5 מושבים במחיר המבוקש, זה אפשרי.

חישוב פונקציית המטרה:

עבור כל סידור ישיבה פוטנציאלי, האלגוריתם מחשב את הערך של פונקציית המטרה 𝐹.

בהינתן שהאולם ריק, ניתן לבחור כל בלוק של 5 מושבים רצופים.

5 המושבים הרצופים שנבחרו מוצגים למשתמש וניתנת לו האפשרות לבחור בסידור הישיבה או לסרב לקבלו. במידה והמשתמש בוחר לקבל את ההצעה, מוקצים למשפחה ופריסת האולם מעודכנת כדי לשקף את ההקצאה.

על מנת להדגים את תסריט 1, נציג את מפת האולם לפני אתחול מסד הנתונים ולאחר שיצרנו סרט חדש וביקשנו לבצע הזמנה לפי תיאור תסריט-

יצירת אירוע חדש (לאחר שלב זה, האירוע נשמר במערכת ולמשתמש יש אופציה לבצע הזמנה לאירוע שנוסף בהצלחה)

A screenshot of a computer

Description automatically generated

הגדרת תצורת האולם (מחיר 0 משמע שאין כיסא באולם ולכן בדרך זאת ניתן להגדיר מפות אולם בתצורות שונות)-

A screenshot of a calculator

Description automatically generated

האירוע נוסף בהצלחה ולמשתמש מוצג על המסך כי הצליח-->

A green text on a white background

Description automatically generated

נראה כי האולם ריק בשלב זה (נבחר באירוע 22)

A screenshot of a computer

Description automatically generated

כל המושבים זמינים כי עדיין לא בוצעו הזמנות לאולם (לכן המושבים מסומנים בירוק, אם עומדים על אחד המושבים עם העכבר ניתן לקבל את המחיר עבור כל כיסא)

A screenshot of a computer screen

Description automatically generated

יצירת הזמנה לאירוע ע"י המשתמש ע"י לחיצה על הכפתור Make Invitation to event (במסך זה, המשתמש בוחר את האירוע אליו מעוניין ליצור הזמנה חדשה. האירועים המוצגים הם האירועים המופיעים במערכת).

A screenshot of a computer

Description automatically generated

המשתמש מקליד כי מעוניין בהזמנת 5 מושבים רצופים במחיר 5$ כל מושב-

A screenshot of a invitation

Description automatically generated

למשתמש מוצג המושבים אותם בחר עבורו האלגוריתם וניתנת למשתמש הבחירה האם לקבל את ההושבה או לא לסרב לקבל את ההושבה.

A screenshot of a invitation

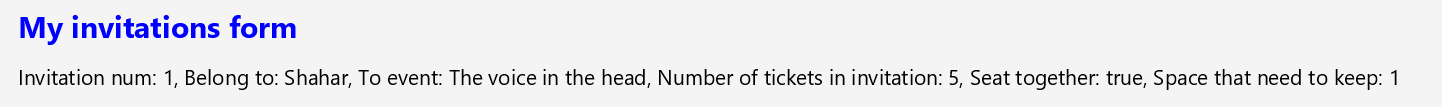
Description automatically generated

המשתמש קיבל הסידור->

A green text on a white background

Description automatically generated

למשתמש מוצגת הזמנתו ע"י לחיצה על הכפתור Show my invitations-



נציג את מפת האולם בהנחה והמשתמש בחר לקבל את סידור ההושבה שהתקבל ע"י המערכת-

A screenshot of a event seating form

Description automatically generated

ניתן לראות כי המושבים בשורה הראשונה (אותם האלגוריתם בחר כסידור הושבה עבור המשתמש והמשתמש קיבל את הסידור) סומנו בצבע אדום- משמע, המושבים תפוסים/מוקצים עבור המשתמש ולא ניתן להשתמש במושבים אלה עבור הזמנה אחרת. מאחר והמושבים שנבחרו הם עבור אנשים בריאים, נשמר מרחק של 1 מכל הזמנה נוספת (לכן המושבים בשורה 2 מסומנים באפור).

מבחינת הconsole –

A computer screen shot of a computer code

Description automatically generated

ניתן לראות שהערך הנמוך ביותר של פונקציית המטרה מתקבל עבור שורות 1 ו5 ולכן האלגוריתם בחר בשורה 1 שזה הסידור הראשון שמצא.

ננסה כעת להזמין 5 מושבים נוספים,

נבחר שוב בסרט- "הקול בראש" עם id=22

A screenshot of a computer

Description automatically generated

נבחר 5 מושבים במחיר 5$ כ"א-

A screenshot of a invitation

Description automatically generated

האלגוריתם הציע למשתמש את שורה 3-

A screenshot of a invitation

Description automatically generated

נבחר לקבל את ההושבה-

A green text on a white background

Description automatically generated

נבחר באולם id=22 כדי להציג את פריסת האולם לאחר קבלת ההושבה-

A screenshot of a computer

Description automatically generated

נציג את פריסת האולם כעת-

A screenshot of a calendar

Description automatically generated

מאחר והאנשים בהזמנה הראשונה והשנייה שביצענו בריאים, האלגוריתם שמר מסביבם מרחק של מקום אחד, לכן המושבים בשורות 2 ו4 לא זמינים יותר להזמנה (נותרה רק שורה 5).

מבחינת הconsloe –

A close-up of a computer code

Description automatically generated

האלגוריתם התחיל את החיפוש שלו החל משורה 3 (מאחר שהשורה הראשונה אוישה כולה באנשים בריאים וצריך לשמור מהם מרחק 1 מכל הזמנה נוספת). האלגוריתם חיפש את הערך שבה פונקציית המטרה מקבלת ערך מינימלי והוא התקבל עבור שורה 3 (value = 230 ) ועבור שורה 5. מאחר והערך הראשוני התקבל עבור שורה 3, נבחרה שורה זו להושבה.

כעת, רק שורה 5 פנויה להושבה.

**תסריט 2- משפחה המבקשת להזמין 5 מושבים רציפים במחיר 5$ אבל אין באולם 5 מושבים ברמת המחיר הזו-**

פעולת משתמש: מגיעה משפחה ומבקשת לקנות 5 מושבים.

תגובת המערכת: המערכת בודקת זמינות של 5 מושבים העומדים במגבלות המרחק.

מטרת האלגוריתם היא להקצות 5 מושבים רצופים ב-$5 כל אחד, או לספק אפשרויות ישיבה חלופיות אם המושבים המבוקשים אינם זמינים. הוא נותן עדיפות לישיבה יחד, ואם זה לא אפשרי, הוא מנסה לפצל את המושבים תוך עמידה בהעדפת המחיר של המשתמש או מציאת מושבים במחיר שונה זה מזה.

שלב 1- פתרון חמדני:

בקשת משתמש: המשתמש מבקש 5 מושבים רצופים ב-$5 כל אחד.

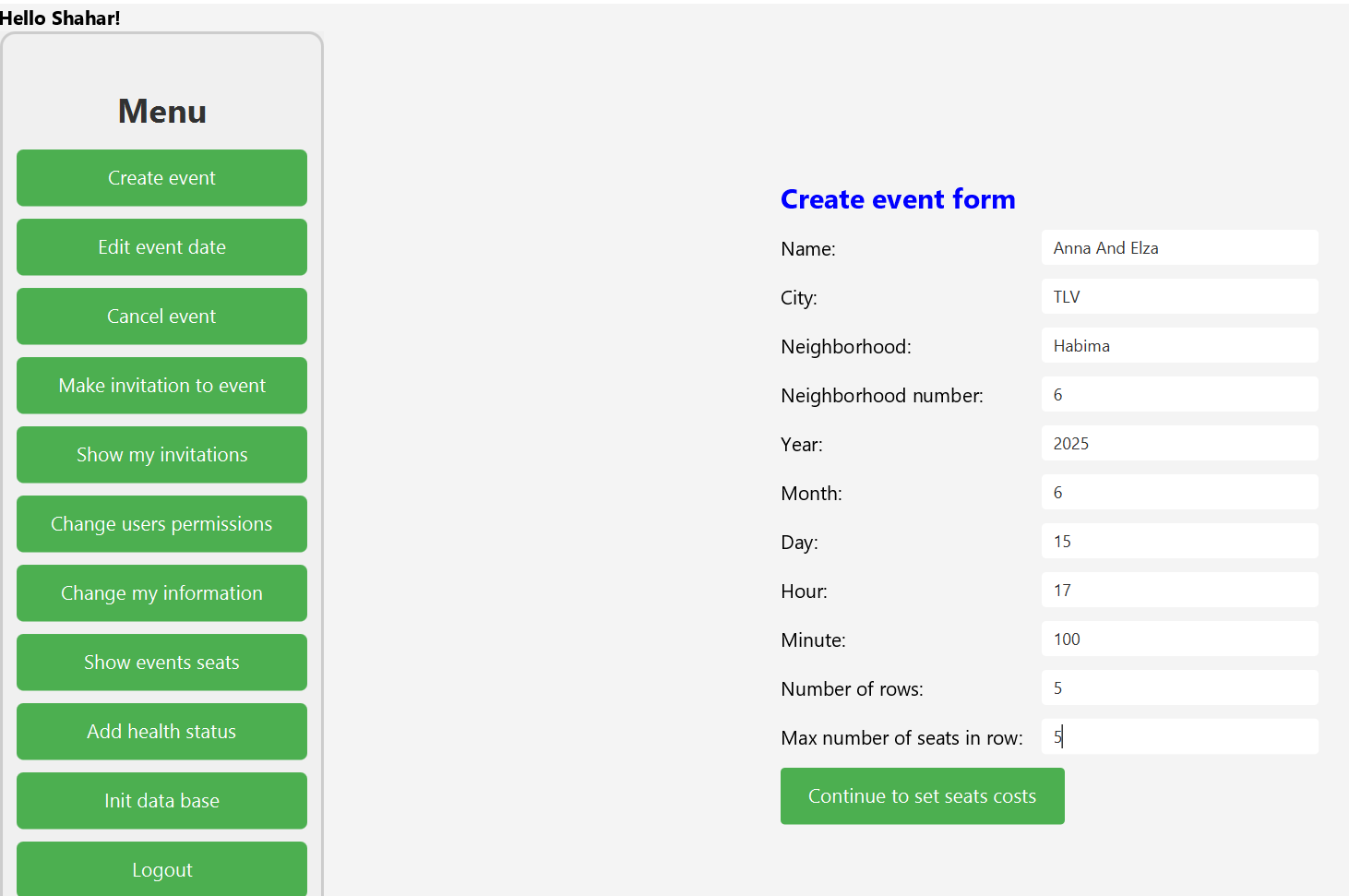
האלגוריתם מתחיל בניסיון למצוא 5 מושבים רצופים ב-$5 כל אחד. הוא סורק את פריסת הישיבה כדי למצוא מושבים פנויים ביחד במחיר המבוקש. אם האלגוריתם מצא 5 מושבים במחיר של 5$ כל מושב, האלגוריתם מסמן את המושבים וממשיך לשלב האופטימיזציה. במידה והאלגוריתם לא מצא 5 מושבים ביחד במחיר המבוקש, האלגוריתם מחפש מושבים בנפרד במחיר המבוקש. אם האלגוריתם לא מצא מושבים ביחד ובנפרד במחיר המבוקש, האלגוריתם מחפש מושבים ביחד במחיר משתנה. במידה והאלגוריתם לא מצא מושבים ביחד במחיר משתנה הוא מחפש מושבים בנפרד במחיר משתנה. במידה והאלגוריתם לא מצא, תוצג הודעה כי אין אפשרות למצוא מושבים העומדים באילוצים.

שלב 2- שלב האופטימיזציה-

לאחר שהאלגוריתם מצא סידור ישיבה אופציונלי חמדני עבור ההושבה, המערכת עוברת לשלב האופטימיזציה בה היא מנסה לשפר ולייעל את הפתרון. האלגוריתם מנסה למצוא מושבים עדיפים יותר שעונים לאילוצים והוא עושה זאת ע"י חיפוש במרחק 10 כיסאות מהפתרון החמדני. השאיפה של האלגוריתם היא למצוא פתרון בעל ערך מינימלי ביותר של פונקציית המטרה, בדרך זאת, המערכת תמקסם את הרווחים עבור בעל האולם, שזה למעשה המטרה העיקרית של המערכת.

על מנת להדגים את תסריט 2, נציג את מפת האולם לפני אחרי שיצרנו סרט (אירוע) חדש וביקשנו לבצע הזמנה לפי תיאור תסריט-

יצירת אירוע חדש-



הגדרת תצורת האולם-

A screenshot of a computer

Description automatically generated

האירוע נוסף בהצלחה-

A green text on a white background

Description automatically generated

נציג את מפת האולם לפני שבוצעה הזמנה (נבחר באירוע 23, אותו הוספנו כרגע למערכת)

A close-up of a message

Description automatically generated

מאחר ועדיין לא בוצעה הזמנה לאירוע 23, כל המושבים מסומנים בירוק (כלומר, זמינים)-

A screenshot of a event seating form

Description automatically generated

המשתמש מעוניין לבצע הזמנה עבור אירוע 23-

A screenshot of a computer

Description automatically generated

המשתמש מעוניין לקבל 5 מושבים במחיר של 5$ כל מושב עבור אירוע 23-

A screenshot of a invitation

Description automatically generated

המערכת מציעה למשתמש אופציה אחרת ממה שביקש. אמנם 5 מושבים רצופים אבל במחיר אחר ולמשתמש ניתנת האופציה לקבל את ההזמנה או לסרב לקבלה.

A screenshot of a invitation

Description automatically generated

המשתמש בוחר לקבל את הצעת המערכת-

A green text on a white background

Description automatically generated

למשתמש ניתנת גם האופציה לצפות בהזמנותיו (ניתן לראות כי עבור הסרט The voice in the head בוצעו 2 הזמנות)

A black text on a white background

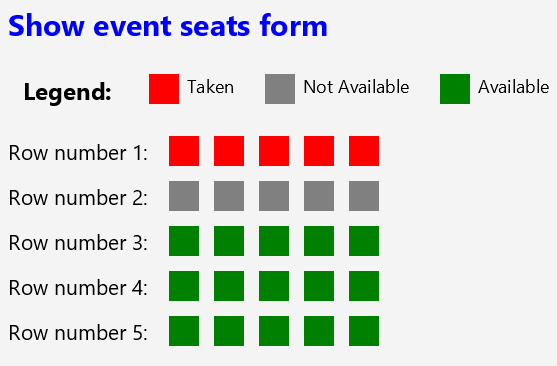
Description automatically generated

נצפה במפת האולם לאחר ההזמנה-

A close-up of a message

Description automatically generated

מפת האולם לאחר הקצאת המקומות עבור המשתמש-



המושבים האדומים מציגים את הזמנת המשתמש, כלומר, תפוסים עבור המשתמשת שחר ולא ניתנים להזמנה עבור הזמנה אחרת (כלומר, תפוסים/מוקצים). שורה 2 מסומנת באפור מאחר ובמושבים של שורה 1 יושבים אנשים בריאים מהם צריך לשמור מרחק של כיסא1 .

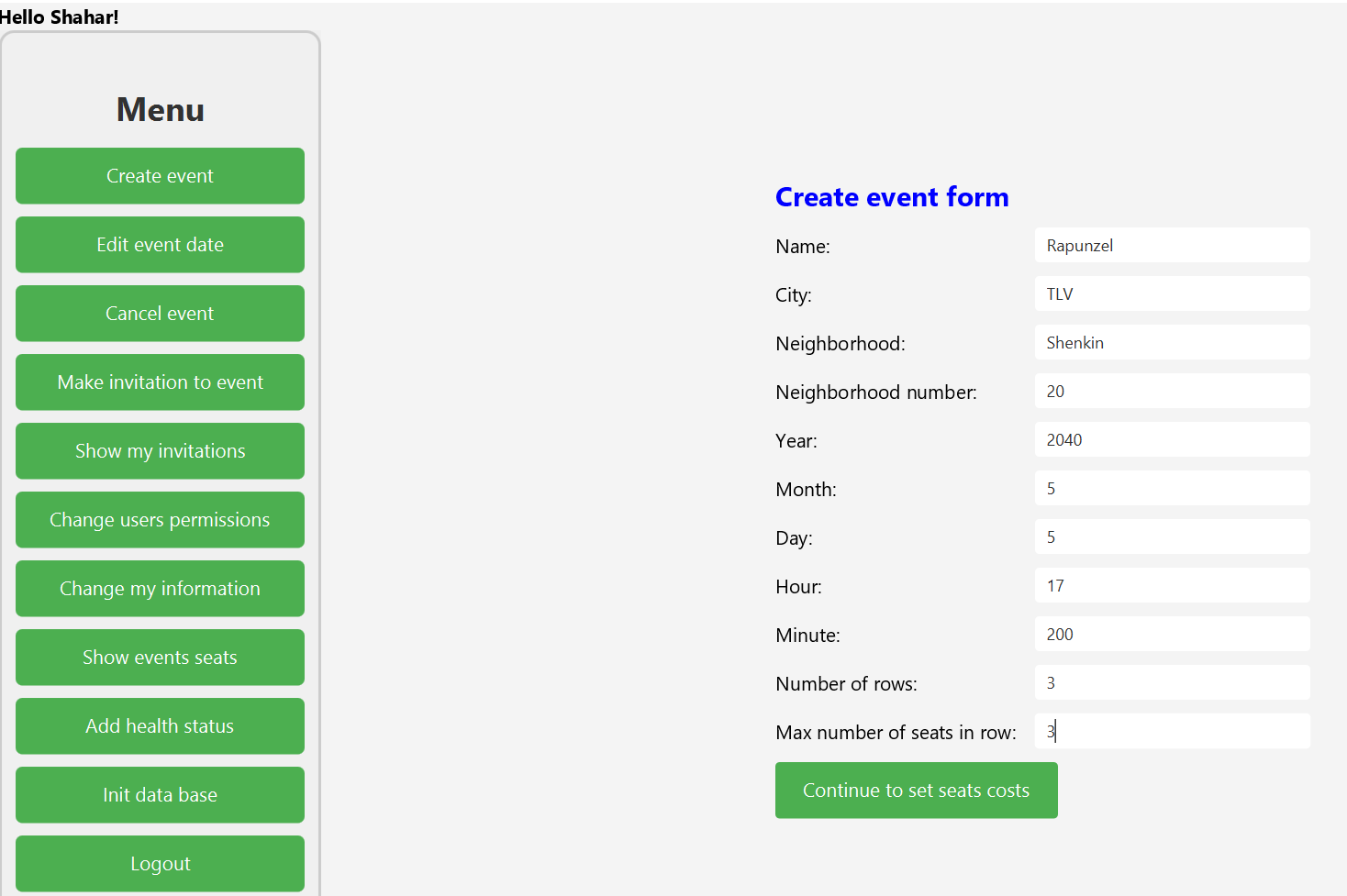
נראה את הconsloe-

A computer screen shot of a computer program

Description automatically generated

נציג את אופן פעולת האלגוריתם עבור אולם קטן יותר בהם אין 5 מושבים רצופים במחיר המבוקש-

נגדיר אירוע חדש-



לפי התסריט, באולם אין מושבים ברמת המחיר 5$ ולכן נגדיר עבור תצורת האולם מחירים שונים-

A screenshot of a computer

Description automatically generated

האירוע נוסף בהצלחה-

A green text on a white background

Description automatically generated

מפת האולם לפני הזמנה (אולם ריק)-

A screenshot of a computer

Description automatically generated

מאחר ועדיין לא בוצעה הזמנה לאירוע 24, כל המושבים מסומנים בירוק (כלומר, זמינים)-

A screenshot of a computer screen

Description automatically generated

נבצע הזמנה לאירוע החדש שנוסף (id=24)-

A screenshot of a computer

Description automatically generated

נבחר 5 מושבים במחיר 5$ כ"א-

A screenshot of a invitation

Description automatically generated

המערכת מציעה למשתמש לפצל את הזמנתו מאחר ואין מושבים ברמת המחיר הנ"ל ופריסת האולם לא מאפשרת 5 מושבים רצופים במחיר 5$ כ"א (האולם הוא 3 על 3).

A screen shot of a invitation

Description automatically generated

בהנחה והמשתמש בוחר לקבל את ההזמנה נציג את פריסת האולם-

A green text on a white background

Description automatically generated

נבחר באירוע המבוקש- (id=24)

A screenshot of a computer

Description automatically generated

נציג את הפריסה-

A screenshot of a computer screen

Description automatically generated

נציג את הconsole-

A screenshot of a computer code

Description automatically generated

האלגוריתם ניסה בתחילה למצוא מקומות רצופים במחיר המבוקש -> אין באולם מקומות כאלה -> האלגוריתם עבר לחפש מקומות מופרדים במחיר המבוקש -> אין באולם מקומות כאלה (אין באולם כיסאות עם מחיר של 5$ כלל) -> האלגוריתם ניסה לחפש מקומות רצופים באותו המחיר -> אין באולם מקומות כאלה (האולם מוגדר כמטריצה 3 על 3) -> האלגוריתם עבר לחפש מקומות מופרדים במחיר כלשהו. האלגוריתם ניסה למצוא מקומות שגורמים לפונקציית המטרה להיות הנמוכה ביותר. לבסוף האלגוריתם בחר במקומות הנ"ל מאחר ששם פונקציה המטרה מניבה ערך מינימלי.

**תרחיש 3- משפחה המבקשת לקנות 5 מושבים במחיר 5$ כל מושב אבל אין 5 מושבים זמינים שלא פוגשים את אילוצי הריחוק החברתי-**

פעולת משתמש: משפחה מגיעה ומבקשת לקנות 5 כרטיסים/מושבים ב-$5 כל אחד.

תגובת המערכת: המערכת בודקת זמינות של 5 מושבים העומדים באילוצי הריחוק החברתי.

אילוצים: בשל דרישות הריחוק החברתי, יש להפריד בין המושבים לפי הנחיות בריאות (עבור מחלת הקורונה מדובר על 5 מושבים), מה שעלול להגביל את הזמינות של מושבים סמוכים שיכולים להיענות לבקשה.

מבחינת אלגוריתם:

האלגוריתם מנסה תחילה למצוא 5 מושבים רצופים במחיר המבוקש של $5 כל אחד העומדים באילוצי ריחוק חברתי.

אם לא נמצאו מושבים כאלה, האלגוריתם מחפש 5 מושבים נפרדים ב-$5 כל אחד שעומדים באילוצים של ריחוק חברתי.

אם אף אחת מהאפשרויות לעיל אינה זמינה, האלגוריתם בודק 5 מושבים רצופים במחיר משתנה התואם את האילוצים.

אם לא נמצאו מושבים רצופים במחיר משתנה, האלגוריתם מחפש 5 מושבים נפרדים במחיר משתנה.

אם לא נמצאו אפשרויות מתאימות, המערכת מודיעה למשתמש כי לא ניתן למצוא מושבים העומדים באילוצים.

שלב 1 – פתרון חמדני-

בקשת משתמש: המשתמש מבקש 5 מושבים ב-$5 כל אחד דרך ממשק המשתמש מבוסס JavaFX.

האלגוריתם מתחיל בסריקת פריסת הישיבה כדי למצוא 5 מושבים רצופים ב-$5 כל אחד שעומדים באילוצי הריחוק החברתי. אם האלגוריתם מוצא 5 מושבים רצופים ב-$5 כל אחד שעומדים באילוצים, הוא מקצה את המושבים וממשיך לשלב האופטימיזציה.

אם לא נמצאו מושבים כאלה, האלגוריתם עובר לחפש מושבים נפרדים ב-$5: אם האלגוריתם מוצא 5 מושבים נפרדים ב-$5 כל אחד העומדים באילוצים, הוא מקצה את המושבים וממשיך לשלב האופטימיזציה.

אם לא נמצאו מושבים כאלה, האלגוריתם עובר לחפש מושבים רצופים במחיר משתנה, אם האלגוריתם מוצא 5 מושבים רצופים במחיר משתנה העומדים באילוצים, הוא שומר את המושבים וממשיך לשלב האופטימיזציה.

אם לא נמצאו מושבים כאלה, האלגוריתם עובר לחפש מושבים נפרדים במחיר משתנה, אם האלגוריתם מוצא 5 מושבים נפרדים במחיר משתנה העומדים באילוצים, הוא שומר את המושבים וממשיך לשלב האופטימיזציה.

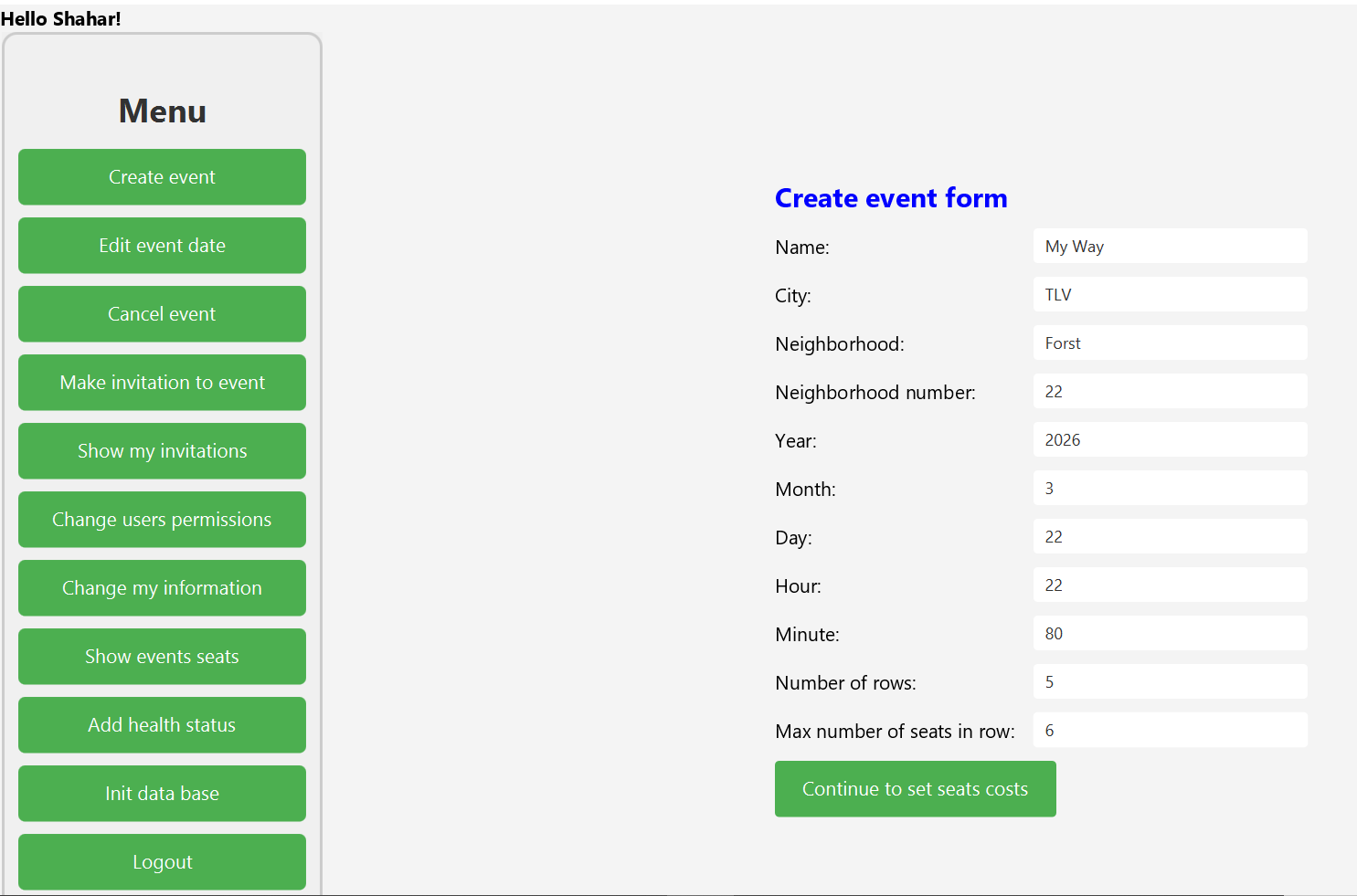
אם לא נמצאו מושבים כאלה, האלגוריתם מודיע למשתמש שלא ניתן למצוא מושבים העומדים באילוצים.

שלב 2 - שלב אופטימיזציה

לאחר שהאלגוריתם מצא סידור ישיבה אופציונלי חמדני עבור ההושבה, המערכת עוברת לשלב האופטימיזציה בה היא מנסה לשפר ולייעל את הפתרון. האלגוריתם מנסה למצוא מושבים עדיפים יותר שעונים לאילוצים והוא עושה זאת ע"י חיפוש במרחק 10 כיסאות מהפתרון החמדני. השאיפה של האלגוריתם היא למצוא פתרון בעל ערך מינימלי ביותר של פונקציית המטרה, בדרך זאת, המערכת תמקסם את הרווחים עבור בעל האולם, שזה למעשה המטרה העיקרית של המערכת.

נדגים את אופן פעולת האלגוריתם עבור תסריט 3-

ניצור אירוע חדש במערכת-



הגדרת מפת האולם-

A screenshot of a computer

Description automatically generated

המשתמש שיצר את האירוע מקבל אישור על כך שהאירוע נוסף למסד הנתונים וכעת ניתן לבצע על אירוע הזמנות-

Green text on a white background

Description automatically generated

נראה את תצוגת הפריסה עבור האירוע שנוסף כרגע-

A screenshot of a computer

Description automatically generated

הפריסה-

A screenshot of a event seating form

Description automatically generated

כעת ננסה לבצע הזמנה לאירוע-

נבחר את האירוע עם id=25 -

A screenshot of a computer

Description automatically generated

נתפוס תחילה 3 מושבים במחיר 10$ כ"א (נרצה להדגים בהמשך כי לאחר תפיסה של 3 מושבים במחיר 10$ כל מושב, לא נוכל להזמין 5 מושבים במחיר 5$ כ"א)-

A screenshot of a invitation

Description automatically generated

ההצעה שמצא האלגוריתם מוצגת למשתמש וניתנת למשתמש הבחירה האם לבחור את ההזמנה או לסרב לקבלה-

A screenshot of a website

Description automatically generated

המשתמש מקבל את ההזמנה מקבל אישור על תפיסת המושבים-

A green text on a white background

Description automatically generated

נציג כעת את מפת האולם העדכנית לאחר תפיסת 3 המושבים במחיר 10$ כל מושב-

נבחר באירוע עם id=25 שכן זהו האירוע אליו ביצענו את ההזמנה-

A screenshot of a computer

Description automatically generated

מפת האולם עבור אירוע id =25 המושבים המסומנים באדום מסמנים את תפיסת המושבים ע"י המשתמש ולכן לא ניתן לתפוס את מושבים אלה עבור הזמנות אחרות (האלגוריתם לא ירוץ על המושבים התפוסים אלא ינסה לתפוס מושבים פנויים אחרים בהתאם לאילוצים). מאחר והמושבים האדומים הוקצו לאנשים בריאים, האלגוריתם שומר מקום 1 מכל מושב ולכן המקומות מסביב למקומות האדומים נצבעו בצבע אפור, כלומר, אי אפשר להושיב שם הזמנה נוספת.

A screenshot of a calendar

Description automatically generated

ננסה כעת לבצע הזמנה של 5 מושבים במחיר 5$ כ"א כדי להדגים את אופן פעולת האלגוריתם-

נבחר שוב באולם id=25 .

A screenshot of a computer

Description automatically generated

נבחר ב5 מושבים במחיר 5$ כ"א, אבל בשביל להראות שלא ניתן למצוא מושבים אלה עבור מפת האולם העדכנית (אין מושבים שלא פוגשים את אילוצי הריחוק החברתי), כלומר, נבחר את המושבים כמושבים בהם צפויים לשבת אנשים חולים-

A screenshot of a invitation

Description automatically generated

המערכת תציג למשתמש כי לא ניתן לבצע את ההזמנה-

Red text on a white background

Description automatically generated

**תרחיש 4- הדגמת האופטימיזציה**

עבור תרחיש 2- משפחה המבקשת להזמין 5 מושבים רציפים במחיר 5$ אבל אין באולם 5 מושבים ברמת המחיר הזו- נרצה להראות שבשביל האופטימיזציה אכן מצאנו מקומות טובים יותר, כלומר, מצאנו מקומות שממקסמים את האילוצים.

המערכת בודקת זמינות של 5 מושבים רצופים ב-$5 כל אחד.

מאחר וחלק מהמושבים כבר נמכרו, המערכת לא יכולה למצוא 5 מושבים רצופים במחיר המבוקש.

האלגוריתם מחפש 5 מושבים ברמת המחיר שהמשתמש הקליד, בנפרד. במידה והאלגוריתם מצא 5 מושבים ברמת המחיר שהוקלדה אז הוא יציע את המושבים למשתמש. במידה והוא לא מצא, האלגוריתם יחפש 5 מושבים ברמת מחיר שונה (מושבים רצופים). במידה והאלגוריתם מצא פתרון למשתמש, הוא יציע את המושבים ולמשתמש תינתן האפשרות לקבל את ההצעה או לסרבה. במידה והאלגוריתם לא מצא, הוא יחפש 5 מושבים ברמת מחיר שונה (מושבים לא רצופים), במידה ונמצאו מושבים כאלה, ההצעה תוגש למשתמש והוא יוכל לקבל את ההצעה או לסרבה. במידה ולא נמצאו מושבים כלל, המערכת תציג למשתמש כי לא ניתן למצוא הזמנה לכמות המושבים.

במהלך שלב האופטימיזציה, האלגוריתם מוצא מקומות טובים יותר הממקסמים את האילוצים.

למשתמש מוצגות אפשרויות מותאמות ויכול לבחור להמשיך עם אחת מהן. אם המשתמש יקבל את הצעות שמציע האלגוריתם, המושבים יישמרו בהתאם. אם המשתמש יסרב, תהליך ההזמנה יופסק.

בתרחיש זה, אנחנו מראים שבשלב האופטימיזציה מצאנו מקומות טובים יותר -> הוצג באופן מפורט בתרחישים הקודמים.

**2.6 שיטות וכלים/פלטפורמות שמומשו במסגרת תכן ופיתוח המערכת-**

כדי לעצב ולפתח בהצלחה את מערכת הקצאת המושבים החכמה, יושם שילוב של כלים הנדסיים מודרניים, טכנולוגיות, פלטפורמות ושיטות. להלן פירוט מפורט של הכלים והשיטות המיושמים בפועל.

כלים ושיטות הנדסיות-

Java - שפת התכנות המשמשת עבור הלוגיקה. השתמשנו ב-Java כדי ליישם את ההיגיון המרכזי של המערכת, כולל האלגוריתמים להקצאת מושבים ואופטימיזציה.

(Cascading Style Sheets) CSS -משמש לתיאור הצגת מסמך שנכתב ב-HTML או XML. תכננו את ממשק המשתמש עם CSS כדי להבטיח שהוא מושך, ויזואלי ועקבי. נעשה שימוש ב-CSS כדי לסגנן את האלמנטים של ה-GUI שנבנה עם JavaFX, תוך שיפור חווית המשתמש.

Eclipse IDE -סביבת פיתוח משולבת המשמשת לכתיבה, בדיקה וניפוי באגים בקוד Java.

Javafx - השתמשנו ב-JavaFX כדי לפתח את ממשק המשתמש הגרפי (GUI) של המערכת. זה אפשר לנו ליצור ממשק אינטראקטיבי וחזותי למשתמשים להזמנת מקומות.

MySQL – ניהול DB - אחסון וניהול נתונים מתמשכים כגון פרופילים של משתמשים, זמינות מושבים, הזמנות ופרטי אירועים.

MySQL Connector/J: מנהל התקן JDBC לחיבור יישומי Java למסדי נתונים של MySQL.

PlantUML - כלי ליצירת דיאגרמות UML - עיצוב ותיעוד ארכיטקטורת מערכת ותרשימי זרימה.

Junit - מסגרת לכתיבת והפעלת מבחני יחידה. כתיבת מבחנים לרכיבים ושירותים בודדים. הבטחת איכות ותקינות הקוד.

Git: מערכת בקרת גרסאות למעקב אחר שינויים בקוד המקור.

GitHub: פלטפורמה לאירוח מאגרי Git והקלת שיתוף פעולה.

מערכת הקצאת המושבים תוכננה ופותחה תוך שימוש בשילוב של כלים הנדסיים, טכנולוגיות, פלטפורמות ושיטות. כלים ושיטות אלו נבחרו ויושמו בפועל על מנת להבטיח פיתוח יעיל, פריסה ותחזוקה של המערכת. הפירוט המפורט שסופק לעיל מדגיש את הטכנולוגיות והפלטפורמות העיקריות בהן נעשה שימוש, מה שמבטיח הבנה מקיפה של הכלים והשיטות ההנדסיות המיושמות במסגרת הפרויקט.

הצגת הכלים והשיטות בטבלה, כנדרש:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| קטגוריה | Tool/Method | תיאור | שימוש בארכיטקטורת המערכת | קישור לארכיטקטורת המערכת |
| Programming Language | Java | שפת התכנות איתה נבנתה המערכת | Developing backend logic and services | Used in business logic, service classes, and algorithms |
| Development Environment | Eclipse IDE | סביבת פיתוח משולבת | כתיבה, בדיקה ואיתור באגים של הקוד | Main IDE for development |
| UI Framework | JavaFX + CSS | מסגרת לבניית GUI | GUI עיצוב ממשק משתמש להזמנת מושבים וניהול חשבונות ולמעשה כל הממשק איתו המשתמש מבצע אינטראקציה עם המערכת | Implemented in the presentation layer |
| Database | MySQL | מערכת ניהול מסדי נתונים | אחסון וניהול נתונים | משמש בשכבת הנתונים לאחסון פרופילים של משתמשים, זמינות מושבים, הזמנות ואירועים |
| Database Tool | MySQL Workbench | כלי לעיצוב וניהול מסדי נתונים של MySQL | תכנון, בניית מודלים וניהול מסד נתונים | עיצוב וניהול מסדי נתונים |
| JDBC Driver | MySQL Connector/J | מנהל התקן JDBC לחיבור יישומי Java ל-MySQL | קישוריות ותפעול מסדי נתונים | Facilitates communication between the business logic layer and the database |
| Diagram Tool | PlantUML | כלי ליצירת דיאגרמות UML | עיצוב ותיעוד ארכיטקטורת מערכת ותרשימי זרימה | Used for visualizing system architecture and process flow |
| Testing Framework | JUnit 5 | מסגרת לכתיבת והפעלת מבחני יחידה | כתיבת מבחנים לרכיבים ושירותים בודדים | מבטיח איכות ותקינות קוד |
| Version Control | Git | Platform for hosting Git repositories | Hosting code repositories, facilitating collaboration | Repository management and collaboration |
| Repository Hosting | |  | | --- | |  |   GitHub | כלי אינטגרציה | Automating build, test, and deployment processes | מבטיח אספקת תוכנה אוטומטית ואמינה |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| קטגוריה | כלי | תיאור | **שימוש** | **Link to System Architecture** |
| Human-Machine Interfaces (GUI/UI/UX) | JavaFX, Scene Builder | JavaFX היא מסגרת המשמשת לבניית ממשק המשתמש הגרפי, ו- Scene Builder הוא כלי עיצוב חזותי ליצירת פריסות FXML. | עיצוב ויישום ממשק המשתמש להזמנת מושבים וניהול חשבונות, הבטחת חוויה אינטואיטיבית וידידותית למשתמש. | מיושם בשכבת המצגת, ומאפשר אינטראקציות עם המשתמש. |
| Entity Relationship Diagram (ERD) | MySQL Workbench | MySQL Workbench משמש לתכנון וניהול מסדי נתונים של MySQL, כולל יצירת ERDs. | ייצוג חזותי של סכמת מסד הנתונים, הצגת ישויות, תכונות וקשרים. | מספק מבנה ברור לשכבת הנתונים, מבטיח עיצוב וניהול נאותים של מסד הנתונים. |
| Use Scenarios (Use Cases) | PlantUML | PlantUML משמש ליצירת דיאגרמות UML לתכנון ותיעוד מקרי שימוש. | הדמיית אינטראקציות של משתמשים עם המערכת, כגון הזמנת מושבים וניהול חשבונות. | מספק הבנה ברורה של אינטראקציות המשתמש והתנהגות המערכת. |
| Algorithm Specifications and Programming Languages | Java | הקצאת מושבים ואופטימיזציה- האלגוריתם נועד להקצות מושבים למשתמשים תוך מיקסום אילוצים כגון העדפות מחיר, מושבים רציפים ומגבלות מרחק של אילוצי הבריאות. | Implementing business logic, such as seat allocation, optimization, and user management. | מרכיב ליבה בשכבת ההיגיון העסקי, הבטחת תפעול מערכת יעיל ונכון. |

**3 מפרטי תכן וארכיטקטורת מערכת הAlpha שמומשה במסגרת הפרויקט-**

**3.1 תכן מפורט (Low Level) של וארכיטקטורת מערכת הAlpha שיושמה-**

Class name- view (שכבת תצוגת המערכת, Javafx UI)

ממשק המשתמש של JavaFX מספק את ממשק המשתמש הגרפי המאפשר למשתמשים ליצור אינטראקציה עם מערכת Alpha. הוא אחראי על הצגת התפריט הראשי, כניסה, רישום, יצירת אירועים ופונקציות אחרות.

Menu Layout and Buttons : פריסת התפריט מספקת אפשרויות שונות למשתמשים לנווט במערכת ולבצע פעולות שונות.

התפריט נראה כך-



- כפתור התחברות: מתחיל את תהליך הכניסה.

- כפתור הרשמה: מתחיל את תהליך הרישום.

- כפתור יצירת אירוע: מתחיל את תהליך יצירת האירוע.

- כפתור עריכת תאריך אירוע: מתחיל את תהליך עריכת פרטי האירוע.

- כפתור ביטול אירוע: מתחיל את תהליך ביטול האירוע.

- כפתור ביצוע הזמנה לאירוע: מתחיל את תהליך יצירת ההזמנה.

- כפתור המציג את הזמנות המשתמש.

- כפתור המקנה את האפשרות לערוך את הרשאות המשתמשים.

- כפתור המקנה את האפשרות לשנות את פרטי המשתמש (שם, מייל וטלפון).

- כפתור המקנה את האפשרות לראות את פריסות ההושבה בכל אירוע ואת סך המושבים הזמינים, והלא זמינים להושבה (כולל מחיר של כל כיסא המופיע בפריסת האולם).

- כפתור הוספה של עדכון הנחיות הבריאות.

- כפתור המאתחל את מסד הנתונים.

- כפתור יציאה מהמערכת.

Login View- תצוגת הכניסה מאפשרת למשתמשים להזין את האימייל שלהם כדי להיכנס למערכת:

private TextField textFieldOfEmailToLogin

private Button loginButton

Register View- תצוגת הרישום מאפשרת למשתמשים חדשים לספק את הפרטים שלהם ליצירת חשבון:

private TextField textFieldOfNameToRegister, textFieldOfEmailToRegister, textFieldOfPhoneToRegister

private Button registerButton

Create Event View- תצוגת יצירת האירוע מאפשרת למשתמשים ליצור אירוע חדש:

private TextField textFieldOfNameToCreateEvent, textFieldOfCityToCreateEvent, textFieldOfNeighborhoodToCreateEvent, textFieldOfNeighborhoodNumberToCreateEvent,

textFieldOfYearToCreateEvent, textFieldOfMonthToCreateEvent, textFieldOfDayToCreateEvent, textFieldOfHourToCreateEvent,

textFieldOfMinuteToCreateEvent, textFieldOfMaxNumberOfRowsToCreateEvent, textFieldOfNumberOfSeatsInRowToCreateEvent;

private Button continueToSetSeatsCostsButton, createEventButton;

Edit Event Date View- תצוגת עריכת תאריך אירוע מאפשרת למשתמשים לעדכן את התאריך והשעה של אירוע קיים:

private TextField textFieldOfEventIdToEditEventDate, textFieldOfYearToEditEventDate, textFieldOfMonthToEditEventDate, textFieldOfDayToEditEventDate,

textFieldOfHourToEditEventDate, textFieldOfMinuteToEditEventDate;

private Button selectEventToEditDateButton, editEventDateButton;

Cancel Event View- תצוגת ביטול האירוע מאפשרת למשתמשים לבטל אירוע קיים:

private TextField textFieldOfEventIdToCancel;

private Button cancelEventButton;

Make Invitation to Event View- תצוגת ההזמנה לאירוע מאפשרת למשתמשים להזמין אחרים לאירוע על ידי ציון פרטים כגון מזהה האירוע, מספר המושבים, מחיר המושב, מצב הבריאות והאם המושבים צריכים להיות ביחד:

private TextField textFieldOfEventIdToMakeInvitation, textFieldOfNumOfSeatsInInvitationToMakeInvitation, textFieldOfSeatPriceToMakeInvitation;

private ComboBox<String> comboBoxOfHealthStatusToMakeInvitation;

private CheckBox checkBoxOfAllSeatsNeedToBeTogetherToMakeInvitation;

private Button selectEventToMakeInvitationToButton, showOfferSeatsInvitationButton, makeInvitationToEventButton, dontMakeInvitationToEventButton;

Change Users Permissions View- תצוגת שינוי הרשאות משתמשים מאפשרת למנהלי מערכת לעדכן את ההרשאות של משתמשים אחרים:

private TextField textFieldOfEmailToChangeUsersPermissions, textFieldOfPermissionToChangeUsersPermissions;

private Button changeUserPermissionsButton;

Change My Information View- תצוגת שינוי המידע שלי מאפשרת למשתמשים לעדכן את המידע האישי שלהם:

private TextField textFieldOfNameToChangeMyInformation, textFieldOfEmailToChangeMyInformation, textFieldOfPhoneToChangeMyInformation;

private Button changeMyInformationButton;

Show Event Seats View- תצוגת מושבי המופע מאפשרת למשתמשים לצפות בסידור הישיבה לאירוע ספציפי:

private TextField textFieldOfEventIdToShowEventSeats;

private Button showEventSeatsButton;

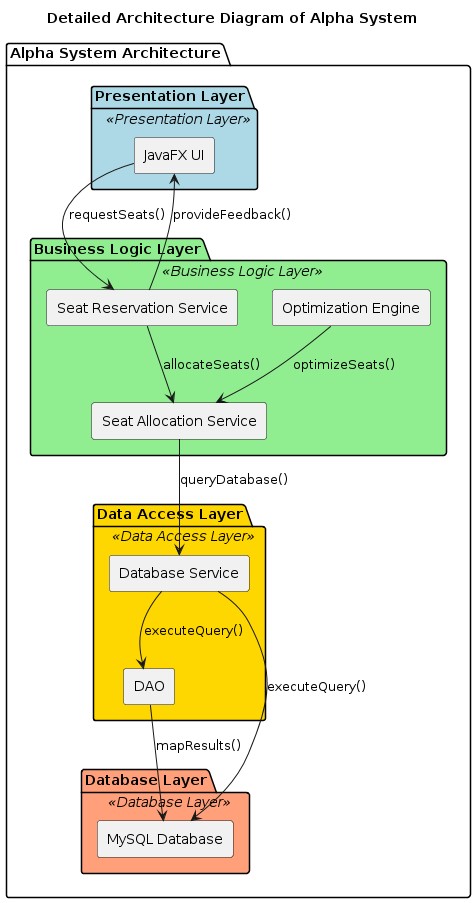
Add Health Status View- תצוגת הוסף מצב בריאות מאפשרת למנהלי מערכת להוסיף מצבי בריאות חדשים למערכת:

private TextField textFieldOfNameToAddHealthStatus, textFieldOfSpaceToAddHealthStatus;

private Button addHealthStatusButton;

Interaction with the System- מחלקת View בממשק המשתמש של JavaFX מספקת את המבנה הבסיסי לכל האינטראקציות של המשתמשים עם מערכת Alpha. כל כפתור ושדה טקסט מייצגים פעולות שונות שמשתמש יכול לבצע, החל מכניסה והרשמה ועד יצירת אירועים וביצוע הזמנות. אינטראקציות אלו מעובדות על ידי השירותים המקבילים בשכבת הלוגיקה העסקית, המטפלים בהקצאת המושבים בפועל, באופטימיזציה ובפעולות מסד הנתונים.

**3.2 תרשים ארכיטקטורה מפורט ברמת בלוקים ומודולים טכנולוגיים של ה Alpha שמומשה במסגרת הפרויקט-**



**3.3 הבלוקים והמודולים במערכת וכיצד משתלבים בתפקוד המערכת ובתרומה ליישום הארכיטקטורה הכוללת של המערכת-**

Javafx UI –

ממשק המשתמש של מערכת ה Alpha, שנבנה באמצעות JavaFX. זה מאפשר למשתמשים ליצור אינטראקציה עם המערכת על ידי מתן קלט וקבלת משוב.

מטפל בבקשות משתמשים להזמנת מקומות.

מציג אפשרויות ישיבה זמינות למשתמשים.

מקל על אישור המשתמש של הזמנת מקומות.

מכיל את כלל כפתורי התפריט באמצעותם המשתמש מנווט בתוך המערכת.

תרומה: מספק ממשק אינטואיטיבי ואינטראקטיבי למשתמשים, המבטיח קלות שימוש וחוויה חלקה. שכבה זו חיונית ללכידת קלט המשתמש והצגת מידע רלוונטי, המשפיעה ישירות על שביעות רצון המשתמש ואימוץ המשתמשים את המערכת.

Business Logic Layer

Seat Reservation Service

מטפל בבקשות משתמשים להזמנת מושבים, מקיים אינטראקציה עם שירותי הקצאת המושבים והאופטימיזציה, ומספק משוב לממשק המשתמש.

מעבד בקשות משתמשים למושבים.

מספק משוב לממשק המשתמש לגבי זמינות מושבים וסטטוס הזמנה.

תרומה: פועל כמתווך בין ממשק המשתמש ולוגיקת הקצאת המושבים והאופטימיזציה הבסיסית, ומבטיח שבקשות המשתמש מעובדות כהלכה והתגובות מועברות כראוי בחזרה למשתמש.

Seat Allocation Service

מנהל את הקצאת המושבים על סמך בקשות ואילוצים של משתמשים והנחיות הבריאות.

מקצה מושבים לפי זמינות והעדפות המשתמש.

בודק את הזמינות הנוכחית של מושבים.

השירות מבטיח שהמושבים מוקצים ביעילות תוך הקפדה על הנחיות הריחוק החברתי ואילוצי המשתמש. הוא ממלא תפקיד מכריע בהקצאת מושבים בזמן אמת, ומשפיע ישירות על ביצועי המערכת ושביעות רצון המשתמשים.

Optimization Engine

מיישם אלגוריתמים כדי לייעל את הקצאת המושבים, למצוא את סידור הישיבה הטוב ביותר האפשרי שממקסם את האילוצים ואת שביעות רצון המשתמש.

מייעל את ההקצאה הראשונית של המושבים כדי למצוא סידור טוב יותר.

משווה סידורי מושבים שונים כדי לבחור את האופטימלי.

תרומה: משפר את היעילות והאפקטיביות של תהליך הקצאת המושבים על ידי מציאת סידורי הישיבה הטובים ביותר. מודול זה מבטיח שהמערכת ממקסמת את שביעות רצון המשתמש וההכנסה על ידי אופטימיזציה של השימוש במושב.

Data Access Layer

Database Service

מנהל אינטראקציות עם מסד הנתונים של MySQL, כולל פעולות CRUD לזמינות מושבים, פרופילים של משתמשים והזמנות.

מבצע שאילתות לאחזור נתונים.

מעדכן את מסד הנתונים בנתונים חדשים.

תרומה: משמש כגשר בין שכבת ההיגיון העסקי לשכבת מסד הנתונים, ומבטיח אחזור ועדכונים חלקים של נתונים. שירות זה חיוני לשמירה על הדיוק והשלמות של נתוני המערכת.

Data Access Objects (DAOs)

מספק שכבת הפשטה עבור פעולות מסד נתונים, המבטיח שהלוגיקה העסקית יכולה לקיים אינטראקציה עם מסד הנתונים ללא צורך לדעת את פרטיו.

מבצע שאילתות מסד נתונים.

ממפה תוצאות שאילתות למבני נתונים של יישומים.

תרומה: מפשט את האינטראקציה בין ההיגיון העסקי למסד הנתונים, ומאפשרת קוד ניתן לתחזוקה וגמיש יותר. DAOs מבטיחים שפעולות הנתונים מטופלות בצורה יעילה ונכונה.

Database Layer

MySQL Database

מסד הנתונים היחסי המשמש לאחסון כל הנתונים הקבועים, כולל פרופילים של משתמשים, זמינות מושבים, הזמנות ופרטי אירועים.

מאחסן נתונים כגון פרופילים של משתמשים, זמינות מושבים והזמנות.

מאחזר את הנתונים הדרושים לאפליקציה.

תרומה: מספק פתרון אחסון חזק ואמין לנתוני המערכת, המבטיח שכל המידע מאוחסן בצורה מאובטחת וניתן לאחזור בקלות. שכבת מסד הנתונים חיונית לפונקציונליות הכוללת של המערכת, המאפשרת אחסון ושליפה של נתונים קריטיים.

כל בלוק ומודול בארכיטקטורת מערכת אלפא ממלא תפקיד ספציפי בהבטחת תפעול יעיל ואפקטיבי של מערכת הזמנת המושבים. מלכידת קלט המשתמש וניהול הקצאת מושבים ועד לאופטימיזציה של סידורי הישיבה וטיפול באחסון נתונים, כל רכיב תורם לפונקציונליות ולביצועים הכוללים של המערכת. תרשים התכנון והארכיטקטורה המפורטים ממחישים כיצד מרכיבים אלה מתקשרים, ומספקים מבט מקיף על מבנה המערכת ופעולותיה.

**4 ממצאי ביצוע בדיקות מערכתיות**

**4.2 פירוט של ביצוע תוכנית הבדיקות של המערכת וה Alpha, על פי מסגרת התכנון שפורטה בדוח ההנדסי-**

ביצוע תוכנית הבדיקות של המערכת ושל אלפא.

תוכנית הבדיקות של המערכת וגרסת האלפא שלה בוצעה על פי המסגרת התכנונית המפורטת בדוח ההנדסי. חלק זה מתאר את הצעדים המרכזיים שננקטו, שיטות הבדיקה שהופעלו, הממצאים וכל חריגה מהתוצאות הצפויות.

המטרה העיקרית של תוכנית הבדיקות הייתה לוודא שהמערכת פועלת על פי המפרט ועומדת בכל הדרישות הפונקציונליות והלא פונקציונליות.

הטבלה שהוצגה בדוח ההנדסי-

|  |  |  |  |
| --- | --- | --- | --- |
| מודל נבדק | סוגי בדיקות | תוצאה רצויה | תקין/לא תקין |
| DB | 1.שלמות הנתונים  2.זמני גישה לנתונים  3.עדכון נתונים  -נאמת את זרימת הנתונים בין ממשק המשתמש לשכבת business logic layer  -בדיקת אינטגרציה בין שכבת הלוגיקה לבסיס הנתונים. | נרצה שכל הנתונים ישמרו ולא תהיה זליגת נתונים מהמערכת.  נשאף שזמני הגישה לנתונים יהיו נמוכים ככל האפשר על מנת שזמני התגובה של המערכת יהיו נמוכים. | מקרה של זליגת הנתונים הוא מקרה קריטי אשר ישפיע באופן ישיר על יעילות האלגוריתם (שהוא ליבת המערכת)-- > לא תקין.  יעילות האלגוריתם הוא פועל יוצא של זמני הגישה לנתונים (בין היתר). במקרים בהם זמני הגישה לנתונים גבוהים, הדבר ישפיע על ביצועי המערכת אבל לא על הפונקציונליות ולכן נשאף שהזמנים יהיו נמוכים ככל האפשר, אך במקרים בהם לא נצליח להגיע לזמנים נמוכים, נעדיף להשקיע משאבים במקומות אחרים --> תקין (יחסית). |
| Seat Selection and Booking (Optimal Algorithm) | נבדוק את פונקציונליות בחירת המושבים, כולל מקרי קצה, כמו- בחירת מושבים מוזמנים, פיצול הזמנה במקרים בהם אין מושבים סמוכים. | במקרים בהם לא ניתן להציע למשתמש ישיבה במושבים סמוכים, נציע למשתמש לפצל את ההזמנה (ל2 קבוצות או יותר, תלוי במספר המקומות הפנויים באולם). נשאף שגם לאחר פיצול ההזמנה, כמה שיותר מושבים יהיו סמוכים.  בנוסף, נבדוק שהמערכת לא מציעה למשתמש בחירה של מושבים מוזמנים.  נשאף להקצאה מיטבית והוגנת של מושבים על בסיס זמינות, העדפות המשתמש והנחיות הבריאות המשתנות. | המערכת תציע למשתמש כיסאות תפוסים --> לא תקין.  בהנחה וניתן למצוא מקומות סמוכים והמערכת תפצל את ההזמנה כך שכל (או רוב) חברי הקבוצה ישבו במקומות שונים באולם (כמה שפחות קבוצות מפוצלות)-- > לא תקין.  המערכת לא תתחשב בנהלי הבריאות --> לא תקין. |
| Error Handling | נבדוק את התנהגות המערכת בתרחישים שגויים, כמו כשלי רשת או קלט לא חוקי. | נשאף לתפוס את רוב שגיאות המערכת (או לכל הפחות את השגיאות הקריטיות) ונציג אותם למשתמש באמצעות משוב אינטראקטיבי (הודעות שגיאה למסך), נעשה זאת ע"י- בדיקת התנהגות המערכת לאחר עדכונים (למשל, תיקוני תוכנה, תכונות ואובייקטים חדשים). | שגיאות בקריאת הקלט מהמשתמש עלולות להוביל לשגיאות ותעופות בתוכנית ולכן לרוב נתבסס על קלט חוקי מהמשתמש (כמו שעת הסרט) --> תקין. |

בקוד המערכת ישנו package המכיל class-ים שבתוכם הבדיקות האוטומטיות והידניות שהרצנו על התוכנית בהתאם לתרחישים.

**5 דיון (Discussion)-**

**5.1 סיכום המסקנות והלקחים שהופקו-**

הפיתוח והבדיקה של המערכת וגרסת אלפא היו חוויה מקיפה ומעשירה. פרויקט זה כלל מספר שלבים, כולל תכנון, יישום, בדיקה ואופטימיזציה. לאורך השלבים הללו נתקלו באתגרים והישגים שונים, שהובילו לתובנות ולקחים בעלי ערך שלבסוף גרמו לנו ליצור מוצר חדשני ובעל ערך לשוק.

עיקרי המסקנות וההישגים עיקריים כללו-

עיצוב ותכנון יעיל: הפרויקט התחיל בשלב תכנון ותכנון חזק. דיאגרמות ארכיטקטוניות מפורטות, תרחישי מקרה שימוש ותוכניות בדיקה סיפקו בסיס איתן לתהליך הפיתוח. הכנה קפדנית זו אפשרה שלב יישום חלק יותר והבטיחה התאמה ליעדי הפרויקט. כמו כן, המסמך הייזום והמסמך ההנדסי שהגשנו היו עבורנו תשתית חזקה ומקיפה כאשר התחלנו לפתח את הפרויקט.

בדיקות מקיפות: תוכנית הבדיקות המערכתית הייתה נרחבת, התוכנית כיסתה פונקציות ותרחישים שונים. תסריטי הבדיקה תוכננו כדי להבטיח כיסוי יסודי, כולל מקרים רגילים ומקרים קצה. הביצוע המוצלח של בדיקות אלו אימת את החוסן והאמינות של המערכת.

אופטימיזציה של אלגוריתם: אחד המרכיבים הקריטיים של הפרויקט היה אופטימיזציה של אלגוריתם הקצאת המושבים. למעשה, זוהי ליבת המערכת. הפתרונות החמדניים הראשוניים סיפקו נקודת התחלה, בעוד ששלב האופטימיזציה הבטיח את סידורי הישיבה הטובים ביותר עבור הזמנות בהם נמצאו יותר מסידור ישיבה אחד (בגישה החמדנית). גישה זו הדגימה את החשיבות של איזון פתרונות ראשוניים עם אופטימיזציות מעודנות יותר לתוצאות משופרות.

ממשק משתמש וניסיון: יישום ממשק משתמש מבוסס JavaFX הדגיש את החשיבות של עיצוב ידידותי למשתמש. בתחילה, יצרנו את הממשק באופן פחות ויזואלי אך עם התקדמות הפרויקט, הבאנו כי הטמעה של ממשק ויזואלי, אינטראקטיבי ומושך עבור המשתמשים יגרום להם לאמץ את המערכת ולהשתמש בה לאורך זמן. הממשק תוכנן להיות אינטואיטיבי ומגיב, ולספק חוויה חלקה למשתמשים. התמקדות זו בחוויית משתמש הייתה קריטית לאימוץ המערכת בקרב המשתמשים ולהצלחתה.

ניהול מסד הנתונים: הפרויקט כלל פעולות מקיפות של מסד נתונים, כולל הוספה, עדכון, חיפוש והסרה של רשומות. פונקציות הטיפול במסד הנתונים נבדקו בקפדנות, מה שהבטיח שלמות ואמינות הנתונים. ניהול מוצלח של פעולות מסד נתונים מדגיש את החשיבות של שיטות טיפול חזקות בנתונים.

**יעדים שלא הושגו ובעיות עיקריות שנתקלנו בהן בעת הפיתוח-**

תקשורת בין-מודולים: הבטחת תקשורת חלקה בין מודולים שונים של המערכת הייתה מאתגרת. בעוד שהמודולים פעלו כהלכה לחוד, תקשורת בין-מודולים הובילה לפעמים לבעיות. שיפור זרימת המידע בין המודולים הייתה חיונית לקוהרנטיות המערכת הכוללת.

חשיבותו של תכנון מפורט: בדיעבד, השקעת זמן מספקת בשלבים הראשוניים יכלה למנוע בעיות רבות במהלך היישום. במהלך הפיתוח הבנו כי הרעיון לפתח מערכת חזקה להקצאת המושבים שתבצע אופטימיזציה על כל סידור הישיבה (כלומר, לאחר שכל המשתמשים מקבלים אישור על כרטיס) היא לא ישימה בעולם האמיתי. במסמך הייזום ובמסמך ההנדסי תיארנו את המערכת בצורה כזאת שכל משתמש מקבל הקצאה למושב ולאחר שנסגרת ההרשמה, המערכת מבצעת אופטימיזציה על הסידור החמדני (שמתקבל בתחילה על כל האולם). השקעת זמן רב יותר באפיון האלגוריתם יכלה למנוע את שינוי המערכת כולה.

בנוסף לאתגרים הטכניים, הפרויקט הושפע משמעותית מגורמים חיצוניים, בעיקר מהמלחמה. פרוץ המלחמה עוררה קשיים רבים, שכן עמית נקרא לעתים קרובות למילואים. מצב זה חייב אותנו לפעול בלוח זמנים גמיש יותר (תוך אישורים של הגורמים הרלוונטיים), שגרם לשיבושים בהמשכיות תהליך הפיתוח ולהציב אתגרים לוגיסטיים (התקשנו בעיקר למצוא זמנים משותפים בהם שנינו יכולים לשבת יחד על תהליך הפיתוח). הצוות היה צריך להסתגל במהירות לנסיבות המשתנות, לחלק מחדש את המשימות ולשמור על לוחות הזמנים הפרויקט למרות הכישלונות הללו. ניסיון זה הדגיש את החשיבות של חוסן ויכולת הסתגלות בניהול פרויקטים.

**5.2 הצעה לעבודת המשך, על בסיס הפרויקט והמערכת שמומשו-**

בהתבסס על ההישגים והאתגרים בהם נתקלו במהלך פיתוח הפרויקט, זוהו מספר תחומים לעבודת המשך כדי לשפר עוד יותר את יכולות המערכת ולטפל במגבלות שנצפו. ההצעות הבאות מתארות תחומים מרכזיים לפיתוח ושיפור עתידי:

שיפורי מדרגיות- כדי להבטיח שהמערכת תוכל להתמודד עם מערכי נתונים גדולים יותר ותרחישים מורכבים יותר ביעילות, העבודה העתידית צריכה להתמקד באופטימיזציה של מדרגיות המערכת. זה יכול לכלול:

שיפורים במסד נתונים: שיפור מבני מסד נתונים ויעילות שאילתות לטיפול בכמויות גדולות יותר של נתונים בצורה חלקה.

עיבוד מבוזר: הטמעת טכניקות עיבוד מבוזר כדי לפזר את העומס על פני מספר שרתים, תוך הבטחת ביצועים עקביים.

טיפול באילוץ בזמן אמת-

יישום אילוצי זמן אמת, כגון דרישות ריחוק חברתי, מציב אתגרים משמעותיים. עבודה עתידית צריכה לכוון ל:

אופטימיזציה של אילוצים: פיתוח אלגוריתמים יעילים יותר להתמודדות עם אילוצים בזמן אמת מבלי לפגוע בביצועים.

התאמות דינמיות: הטמעת תכונות המאפשרות למערכת להתאים באופן דינמי אילוצים על סמך נתונים בזמן אמת ואינטראקציות משתמש.

שילוב משופר של משוב משתמשים- שילוב משוב משתמשים בזמן אמת במערכת יכול לשפר משמעותית את השימושיות וההיענות.

בנוסף, בהמשך העבודה על הפרויקט היינו רוצים להוסיף גם אופציה לקבל אישור הזמנה למשתמשים במייל/ הודעה לטלפון והתממשקות המערכת לאמצעי תשלום כדי ליצור מערכת מקיפה המכילה את כל הפונקציונליות הנדרשת כאשר רוצים להזמין מקום באולם קולנוע (כל אולם אחר).

**6 סיכום ומסקנות (Summary and conclusions)-**

חלק זה מדגיש את ההישגים, האתגרים והתובנות שהושגו במהלך הפרויקט, ומספק רפלקציה מקיפה על התהליך כולו.

הישגי מפתח-

הטמענו בהצלחה מערכת הזמנת מושבים מורכבת המשלבת אילוצי זמן אמת, כגון ריחוק חברתי, תוך שמירה על ממשקים ידידותיים למשתמש וביצועים יעילים.

הפיתוח והחידוד של אלגוריתמים לטיפול בתרחישי ישיבה שונים, כולל מציאת פתרונות מיטביים וניהול אילוצים בזמן אמת, היו הישגיים והיו נקודות מפתח בעת פיתוח המערכת. אלגוריתמים אלו נבדקו ואושרו באמצעות מקרי בדיקה נרחבים.

הוקמה מסגרת בדיקה חזקה, המכסה מגוון רחב של תרחישים ומבטיחה את האמינות והיציבות של המערכת. בדיקות אוטומטיות וידניות נערכו כדי לאמת פונקציונליות, ביצועים וטיפול בשגיאות.

עיצוב ממוקד משתמש:

המערכת תוכננה תוך התמקדות בשימושיות, ומספקת ממשק אינטואיטיבי למשתמשים לאינטראקציה. משוב משתמשים שולבו ללא הרף כדי לשפר את החוויה הכוללת.

למרות האתגרים שנתקלנו בהם, אנחנו מאמינים מאוד במערכת שבנינו. לדעתנו, למערכת יש את הפונקציונליות הנדרשת כדי להצליח בשוק ולהיות מערכת שתאומץ ע"י משתמשים רבים הבוחרים לצפות בסרט בקולנוע בעידן שלאחר הקורונה.

בנימה אישית, נהנינו מאוד לפתח את הפרויקט ואנחנו מאמינים שזה לא הפרויקט האחרון שנפתח יחד.

**7 מקורות מעודכנים (References)-**

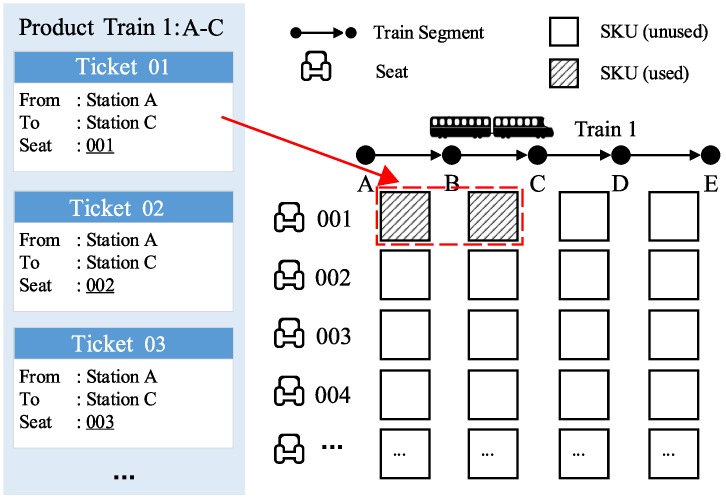
פרק זה מספק עדכון ותיאור מפורט של המקורות המדעיים הרלוונטיים לפרויקט ולמערכת. ההפניות כוללות מאמרים מדעיים, ספרים ואתרים רשמיים או מקצועיים שנחקרו ונלמדו לאורך מחזור החיים של הפרויקט, ממסמך הייזום ועד לספר הפרויקט. הסקר מדגיש מקורות חדשים ונוספים שזוהו במהלך הפרויקט.

מאמרים מדעיים-

Wuyang Yuan & Lei Nie., " Optimization of seat allocation with fixed prices: An application of railway revenue management in China", ,2020.

מאמר זה נועד לייעל את הקצאת המושבים עבור רכבת בסין תוך התחשבות בהתנהגות לקוח. כדי להעריך את ההכנסות ממכירת כרטיסים עם הקצאת מושבים נתונה, מוצע מודל Discrete-Time Markov Chain, התואם למודל ההתנהגות של הלקוחות, כדי לתאר את תהליך הזמנת הכרטיסים תחת בקרה מבוססת מושבים. אלגוריתם גנטי מוצע להתמודד עם המורכבות של מצבי הזמנת כרטיסים. הביצועים של השיטה המוצעת נבדקים בשני ניסויים עם שני אמות מידה אחרות. לאחר מכן אנו מיישמים שיטה זו במקרה של עולם אמיתי עם נתונים מעשיים מקו הרכבת המהירה ננינג-גואנגג'ואו.

שאר המאמר מאורגן באופן הבא: חלק 2 מציג כמה טרמינולוגיות של ניהול הכנסות ומחקרים קודמים על אופטימיזציה של מערכות שונות לניהול הכנסות, כגון מערכות ניהול הכנסות של חברות תעופה בארה"ב. חלק 3 ממחיש את מנגנון הזמנת הכרטיסים בסין. סעיף 4 מגבש את תהליך הזמנת הכרטיסים. סעיף 5 מפתח אלגוריתם גנטי מבוסס סימולציה. סעיף 6 בודק את השיטה על שתי רשתות של רכבות בגדלים שונים ומראה יישום לקו הרכבת המהיר ננינג-גואנגג'ואו. לבסוף, סעיף 7 מסכם את המחקר.



בהקשר של הפרויקט שלנו,

בפרויקט שלנו, האתגר המרכזי היה להקצות מושבים ללקוחות באופן שיעמוד בהנחיות ההתרחקות החברתית תוך ניצול מירבי של השטח הפנוי. האלגוריתמים שתוארו על ידי Smith and Doe סיפק מסגרת לפיתוח הפתרון החמדני הראשוני שלנו ושלב האופטימיזציה שלאחר מכן.

מבחינת האלגוריתם:

פתרון חמדני: הדיון במאמר השפיע על הגישה שלנו למציאת סידור ישיבה ראשוני אפשרי במהירות. הטמענו אלגוריתם חמדני לחיפוש מושבים רצופים ברמת המחיר המבוקשת, מה שמבטיח שהפתרון נמצא ביעילות.

שלב האופטימיזציה: האסטרטגיות המפורטות לאופטימיזציה במאמר הנחו את הפיתוח שלנו בשלב השני של האלגוריתם. השתמשנו בעקרונות המפורטים כדי לחדד את הפתרון הראשוני, תוך התחשבות במספר סידורי ישיבה כדי לבחור את הטוב ביותר בהתבסס על קריטריונים מוגדרים מראש כגון אובדן מינימלי של מושבים ומרחב פנוי מקסימלי.

טיפול באילוצים:

חקירת המאמר בטיפול באילוצים שונים בהקצאת מקומות ישיבה יושמה ישירות על הבעיה שלנו של עמידה בדרישות ריחוק חברתי. השיטות המתוארות אפשרו לנו לשלב אילוצים באלגוריתם שלנו, מה שמבטיח שהפתרונות שסופקו לא רק אופטימליים אלא גם תואמים להנחיות הבריאות.

יעילות אלגוריתם:

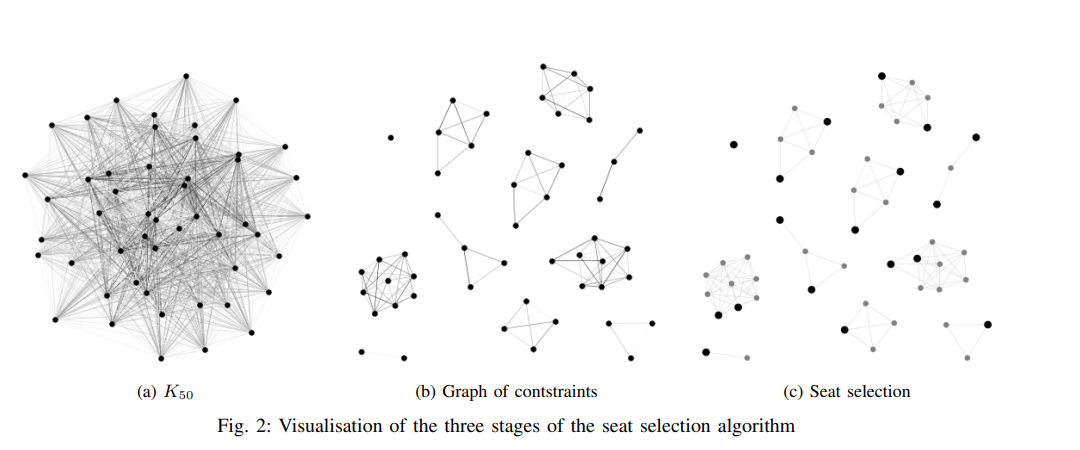
ההתמקדות של Smith and Doe ביעילות של אלגוריתמים להקצאת מושבים עזרה לנו לייעל את ביצועי המערכת שלנו. על ידי אימוץ הפרקטיקות המומלצות שלהם, הצלחנו להבטיח שהפתרון שלנו יהיה יעיל מבחינה חישובית, מה שהופך אותו למתאים ליישומים בזמן אמת בסביבה דינמית.

על ידי מינוף האלגוריתמים והאסטרטגיות המתקדמים המתוארים במאמר של Smith and Doe, הצלחנו לפתח מערכת הקצאת מושבים חזקה ויעילה שעמדה בדרישות הפרויקט והתייחסה למגבלות המוטלות על ידי ריחוק חברתי. התרומות של מאמר זה היו בסיסיות להצלחת הפרויקט שלנו, והיוו בסיס איתן לאסטרטגיות האופטימיזציה שלנו.

Michael Barry, Claudio Gambella, Fabio Lorenzi, John Sheehan & Joern Ploennigs., "Optimal Seat Allocation Under Social Distancing Constraints", IBM Research Europe, 2021.

מגפת Covid-19 הציגה אתגרים ואילוצים חדשים לתכנון עסקי של חזרה לשגרה. המאמר מתאר את בעיית הקצאת השטח המשלבת אילוצי ריחוק חברתי תוך אופטימיזציה של מספר חללי העבודה הבטוחים הזמינים בתרחיש של חזרה לעבודה (שגרה). המאמר מציע ומדגים גישה מבוססת גרפים הפותרת את בעיית האופטימיזציה באמצעות מודלים כגרף דו-צדדי של רכיבים מנותקים על פני גרף של אילוצים. סביבת המשרד, מציגה אתגרים ייחודיים משלה להקצאת מושבים ומגבלות מרחק חברתי. אמנם פריסת משרד עשויה להיות מובנית יותר מזו של תיאטרון, קולנוע או מקום אחר, אך היא גם מגיעה עם מגוון גדול יותר של גדלים וצורות חללים, כמו גם אילוצים תפעוליים אחרים הקשורים לעסקים, כולל תיעדוף צוותים ויחידות עסקיות להקצאת חללי עבודה והרצון לאיתור משותף של צוותים קיימים כך שחברים חוזרים לא יתפרסו על פני מספר קומות או אזורים. כל ארגון מביא מקרי שימוש ייחודיים משלו, מדדי איכות ואילוצים שיש לקחת בחשבון בכל תרגיל הקצאת שטחי עבודה. על אף בעיית הקצאת שטחי העבודה עצמה, החלת מגבלת מרחק חברתי על כל קומה בכל בניין של ארגון גדול היא משימה עצומה. יש לסקור ולהעריך את פריסות הבניינים והמשרדים הקיימים כך שניתן יהיה להשתמש רק באותן חללי עבודה המרוחקים חברתית זה מזה. סקירה, מדידה וסימון של חללי עבודה היא משימה שגוזלת זמן עבור מתכנן שטח או מנהל משרד, במיוחד עבור ארגונים גדולים עם חללי עבודה משרדיים באלפים. יתר על כן, צוות אלה אינם מדעני נתונים או מומחי אופטימיזציה שהוכשרו לנתח את הנתונים ולפתח תוכנית מיטבית.

המאמר מציג גישה אוטומטית לאופטימיזציה של הקצאת שטחי עבודה בסביבות משרדיות בהתבסס על תמונות קיימות של תכנית קומה. במאמר מתוארות גישות שונות לזיהוי חללי עבודה בתמונת קומה. בשלב המתקדם יותר, המאמר מציע גישה מבוססת גרפים לאופטימיזציה של הקצאת סביבת עבודה באמצעות סביבות העבודה שזוהו. לבסוף אנו מציגים גישה מבוססת תוכנית לינארית המשלבת אילוצי מרחק חברתי במודל כללי להקצאת חללי עבודה בסביבת משרד.



A blueprint of a building

Description automatically generated

בהקשר לפרויקט שלנו,

ליבת האלגוריתם של המאמר משלבת אילוצי ריחוק חברתי במודל הקצאת סביבת העבודה. באופן דומה, האלגוריתם שלנו צריך להבטיח שיש מספר מסוים של מושבים בין מושבים תפוסים כדי לעמוד בהנחיות הבריאות. ניתן ליישם ישירות את השיטה המתוארת במאמר לטיפול באילוצים אלו בסביבה משרדית על מסגרת תיאטרון.

תכנות לינארי:

המאמר דן בשימוש בגישה מבוססת תוכנית ליניארית המשלבת אילוצי ריחוק חברתי במודל הכללי להקצאת חללי עבודה. ניתן להתאים מתודולוגיה זו לפרויקט שלנו על ידי ניסוח בעיית תכנות ליניארית שבה הפונקציה האובייקטיבית ממקסמת את תפוסת המושבים תוך כיבוד אילוצי ריחוק חברתי. זה יכול להיות שימושי במיוחד בתרחישים שבהם יש לנו מספר אילוצים שיש לקחת בחשבון, כגון תמחור והעדפות ישיבה קבוצתיות.

אוטומציה ומדרגיות:

הגישה האוטומטית להקצאת שטחי עבודה המוצגת במאמר מדגישה את החשיבות של מדרגיות ויעילות, שהיא חיונית עבור ארגונים גדולים. זה ישים ישירות גם לפרויקט שלנו, שבו האלגוריתם חייב להקצות במהירות וביעילות מושבים בבתי קולנוע בעלי פוטנציאל גדול.

האולם כגרף:

כל מושב בתיאטרון מיוצג כצומת. קצוות מייצגים את אילוצי הריחוק החברתיים, מה שמבטיח שלא ניתן לתפוס את שניהם בצמתים (מושבים) קרובים מדי.

פונקציית המטרה:

פונקציית המטרה באלגוריתם שלנו ממקסמת את הרווחים של בעלי האירוע ואת מספר המושבים המאוכלסים תוך מזעור הקרבה בין המושבים התפוסים והקפדה על ריחוק חברתי.

האילוצים:

אילוצי ריחוק חברתי מעוצבים כדרישות מרחק מינימלי בין מושבים תפוסים. אילוצי תמחור מבטיחים שהמושבים מוקצים על סמך רמות המחיר המבוקשות. העדפות הישיבה הקבוצתיות מנוהלות על ידי הבטחת מוקצים לקבוצות מושבים העומדים הן בדרישות הריחוק החברתי והן בדרישות הקרבה.

שלבים אלגוריתמיים:

קלט: בקשת הישיבה, כולל מספר המקומות והמחיר.

בניית גרף: צור גרף שבו כל מושב הוא צומת, והקצוות מייצגים אילוצי ריחוק חברתי.

אופטימיזציה:

החל תכנות ליניארי כדי למקסם את פונקציית המטרה תחת אילוצים נתונים.

הערך סידורי ישיבה שונים באמצעות פונקציית המטרה.

הפלט- סידור הישיבה האופטימלי העונה על כל האילוצים וממקסם את תפוסת המושבים.

[Malik Braik](https://link.springer.com/article/10.1007/s00521-020-05145-6#auth-Malik-Braik-Aff1), [Alaa Sheta](https://link.springer.com/article/10.1007/s00521-020-05145-6#auth-Alaa-Sheta-Aff2) & [Heba Al-Hiary](https://link.springer.com/article/10.1007/s00521-020-05145-6#auth-Heba-Al_Hiary-Aff1)., "A novel meta-heuristic search algorithm for solving optimization problems: capuchin search algorithm", 2020.

אלגוריתמי חיפוש מטה-יוריסטיים שימשו בהצלחה לפתרון מגוון בעיות בהנדסה, מדע, עסקים ופיננסים. אלגוריתמים מטה-יוריסטיים חולקים תכונות משותפות מכיוון שהם גישות מבוססות אוכלוסייה המשתמשות בסט של פרמטרי כוונון כדי לפתח פתרונות חדשים המבוססים על ההתנהגות הטבעית של יצורים. במאמר זה, הוצג אלגוריתם אופטימיזציית חיפוש חדש בהשראת הטבע הנקרא אלגוריתם חיפוש קפוצ'ין (CapSA) לפתרון בעיות אופטימיזציה מוגבלות וגלובליות. ההשראה העיקרית של CapSA היא ההתנהגות הדינמית של קופי קפוצ'ין. מאפייני האופטימיזציה הבסיסיים של אלגוריתם חדש זה מתוכננים על ידי מודל של פעולות חברתיות של קפוצ'ינים במהלך שיטוט וחיפוש אחר מזון על עצים וגדות נהרות ביערות תוך חיפוש אחר מקורות מזון. חלק מההתנהגויות הנפוצות של קפוצ'ינים במהלך חיפוש מזון המיושמות באלגוריתם זה הן זינוק, נדנדה וטיפוס. קפיצה היא מנגנון יעיל המשמש קפוצ'ינים לקפיצה מעץ לעץ. מנגנוני חיפוש המזון האחרים המופעלים על ידי קפוצ'ינים, הידועים כנדנדה וטיפוס, מאפשרים לקפוצ'ינים לנוע מרחקים קטנים על פני עצים, ענפי עצים וקצוות ענפי העץ. מנגנוני תנועה אלו מובילים בסופו של דבר לפתרונות אפשריים של בעיות אופטימיזציה גלובליות. האלגוריתם המוצע מבוסס על 23 פונקציות בנצ'מרק ידועות, כמו גם פתרון של מספר בעיות הנדסיות מאתגרות ויקרות מבחינה חישובית. מחקר השוואתי רחב נערך כדי להדגים את היעילות של CapSA על פני מספר אלגוריתמים מטה-יוריסטיים בולטים במונחים של דיוק אופטימיזציה וניתוח בדיקות סטטיסטיות. התוצאות הכוללות מראות ש-CapSA נותנת פתרונות מדויקים יותר עם שיעור התכנסות גבוה בהשוואה לשיטות מטה-יוריסטיות תחרותיות.

עבור המערכת שלנו, ניתן להתאים את האלגוריתם כדי לייעל את הקצאת המושבים בבתי הקולנוע על ידי טיפול יעיל באילוצים כגון ריחוק חברתי, מקסום השימוש במושב תוך הקפדה על עמידה בפרוטוקולי הבטיחות. האופי ההיוריסטי של האלגוריתם הופך אותו למתאים לסביבה הדינמית והמוגבלת של הקצאת מושבים.

**8 נספחים (Appendices)-**

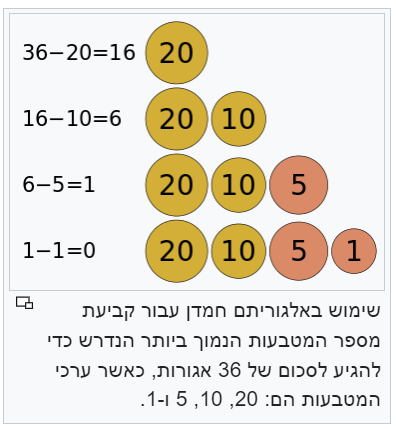
**8.1** נספח פוסטר **הפרויקט-**



**8.5** מידע המסייע לתיאור הפרויקט ולהבנתו-

**אלגוריתם חמדן-**

אלגוריתם חמדן הוא אלגוריתם המתבסס על היוריסטיקה לפיה בוחרים את האפשרות הטובה ביותר הנראית לעין בשלב הנוכחי, מבלי לקחת בחשבון את ההשפעה של צעד זה על המשך הפתרון. אלגוריתמים חמדנים נפוצים בפתרון בעיות מיטוב, בהן מנסים למצוא את הפתרון הטוב ביותר.



דוגמה לאלגוריתם חמדן-

בעיית הסוכן הנוסע: סוכן מכירות רוצה לעבור במספר יישובים כדי למכור את הסחורה שלו. המטרה היא למצוא את המסלול הקצר ביותר שיעבור דרך כל היישובים. על פי שיטת האלגוריתם החמדן, הסוכן הנוסע צריך להסתכל בכל פעם במפה ולנסוע ליישוב הקרוב ביותר בו לא ביקר עדיין. במקרה זה שיטת האלגוריתם החמדן לא תיתן בהכרח את הפתרון הטוב ביותר. כפי שניתן לראות באיור, יכול להיות מצב בו הסוכן ידלג על יישוב מסוים משום שישנו יישוב אחר קרוב יותר, כך שיאלץ לחזור ליישוב עליו דילג בסוף המסלול ולעשות דרך ארוכה יותר.

A diagram of a graph

Description automatically generated

**שיטת חיפוש היוריסטית-**

היא שיטת חיפוש היוריסטית היא שיטת חיפוש המתבטאת בהערכה וניחוש המצמצמים את אזור החיפוש של פתרון מסוים לבעיה ולכן מקצרים את זמן החיפוש אך אינם מבטיחים הצלחה.

בשיטות היוריסטיות מעבר ממצב אחד למצב הבא נקבע על סמך שיקולים היוריסטיים, המציינים את מידת הכדאיות לעבור למצב הבא שנבחר, שממנו ניתן להגיע אל היעד. שיקולים היוריסטיים לא בהכרח מבוססים על חישובים מספריים. את השיקולים היוריסטיים ניתן לייצג בעזרת פונקציות היוריסטיות. פונקציה היוריסטית מתאימה לכל מצב ציון המבטא את ערך השיקולים היוריסטיים, לפיו עוברים למצב הבא.

דוגמאות לשיטות חיפוש היוריסטיות-

Hill-Climbing- זו שיטת חיפוש היוריסטית המבוססת על שיטת חיפוש לעומק. שיטה זו יכולה לקבל עץ קיים או עץ חיפוש הנבנה באופן דינאמי תוך כדי תהליך של פתרון בעיה של מצב במרחב. שורש העץ מייצג את המצב ההתחלתי. לכל צומת מתאימים ציון לפי פונקציית הערכה או פונקציה היוריסטית. ציון הצומת מבטא לרוב את עלות החיפוש ממנו עד צומת המטרה. לפי שיטת החיפוש, מוצאים את הבן בעל הציון הנמוך ביותר של צומת המוצא, מגדירים את הבן כצומת המוצא מחדש וממשיכים ממנו עד שנגיע למטרה או עד שניתקע. החיסרון בשיטה זו הוא האפשרות להגיע למבוי סתום, שהרי רוב הצמתים לא יובילו לפתרון גם אם היה להם את הציון הגבוה ביותר.

Hill Climbing עם Backtracking - זהו הפתרון לחיסרון המתואר לעיל. זה מתבטא בכך שבמידה והמסלול הראשון, הנבנה על פי הצמתים "הטובים ביותר", אינו מוביל לצומת המטרה, מתבצע ניסיון לבנות מסלול נוסף עד קבלת מסלול המוביל לצומת המטרה. זה מתבצע בעזרת שימוש במחסנית.

Beam-Search- שיטת חיפוש זו מבוססת על חיפוש לרוחב. גם בשיטה זאת יש שימוש בפונקציית הערכה שמתאימה ציון לכל צומת. הציון מתאר את סיכוי הצומת להימצא על מסלול המוביל לצומת המטרה. שיטת חיפוש זאת מבוססת על בניית עץ החיפוש לפי רמות. בניגוד לחיפוש לרוחב הרגיל, מפתחים רק את תתי העצים של הצמתים הטובים ביותר בכל רמה. מתקבלת חזית של צמתים בכל שלב, אשר מהם ייבחרו הצמתים להמשך החיפוש לפי טיבן. החיסרון בשיטת חיפוש זו הוא ההתעלמות מצמתים שהיה להם ציון טוב פחות אך הם חלק מהמסלול שמוביל לצומת המטרה. ישנה אפשרות כי הצמתים עם הציון הטוב יותר יובילו למבוי סתום, בעוד מתעלמים מצמתים עם ציון פחות טוב.

Best First- שיטת חיפוש זו משלבת בין חיפוש לעומק וחיפוש לרוחב. בכל שלב בוחרים להמשיך מהצומת המבטיח ביותר בחזית הצמתים ומוצאים את בניו. מצרפים את הבנים לרשימה של צמתים שנבנו עד כה, ושוב בוחרים מהרשימה את הצומת המבטיח ביותר. כך מבצעים עד אשר מגיעים לצומת המטרה.

**אופטימיזציה-**

אוֹפּטִימִיזַצְיָה מתמטית, או מִטּוּב, הוא תת-תחום במתמטיקה שימושית שעוסק במציאת הערך האופטימלי של פונקציה תחת מגבלות אילוצים נתונות. בעיות אופטימיזציה מופיעות בתחומים רבים כגון: מדעי המחשב, הנדסה, חקר ביצועים וכלכלה.

בעיות אופטימיזציה יכולות לעסוק בפונקציות ממשיות, בפונקציות במספר משתנים, בפונקציות מרוכבות, בפונקציונליים וגם בפונקציות שמקבלות ערכים בדידים בלבד.

הקביעה איזה ערך נחשב לאופטימלי תלויה בנתוני הבעיה. בבעיות מסוימות הערך האופטימלי הוא הערך המינימלי שמקבלת הפונקציה הנתונה תחת הגבלות האילוצים, ובבעיות מסוימות זהו הערך המקסימלי. הקושי במציאת הערך האופטימלי תלוי בפונקציה העלות המבוקשת ובתחום בו יש לבצע אופטימיזציה.

