שקף 1:  
**"שלום לכולם.**  
הפרויקט שלנו נקרא **Maze Run**, והוא נעשה כחלק מקורס 'אלגוריתמים מתקדמים למערכות'.  
הרעיון המרכזי היה לקחת בעיה קלאסית – ניווט בתוך מבוך – ולפתור אותה בעזרת אלגוריתמים חכמים.  
אנחנו תכף נציג איך בנינו את זה, מה עשינו מאחורי הקלעים, ואיך נראה המשחק שפיתחנו בסוף.  
העבודה נעשתה בצוות של שלושה – אני, יחד עם \_\_\_ ו־\_\_\_ (להשלים שמות אם רוצים), וכל אחד לקח חלק בפיתוח, באלגוריתמים ובממשק.  
בואו נתחיל."  
  
שקף 2:  
**"אז מה בעצם מימשנו בפרויקט?**  
יצרנו משחק שבו השחקן נמצא בתוך מבוך רנדומלי, והמטרה שלו היא פשוטה – להגיע ליציאה הירוקה, לפני שהאויב האדום תופס אותו.  
אבל בדרך יש גם מכשולים – השחקן צריך לאסוף מטבעות, להתמודד עם אויב אינטיליגנטי, ולהתמודד עם כל מיני הפתעות כמו תור נוסף, רמות קושי משתנות, ואפילו בינה מלאכותית שמתאימה את עצמה למה שהשחקן עושה.

המשחק כולל גם אלמנטים מיוחדים – כמו Power-ups, שלבים קשים יותר, וניקוד.

בקיצור – זה לא רק מבוך. זה מבוך עם חיים."

שקף 3:  
אז קודם כל, השחקן שולט בדמות הכחולה – בעזרת מקשי החצים. הוא יכול לנוע לכל כיוון במבוך, והרעיון הוא לאסוף כמה שיותר מטבעות, להימנע ממכשולים, ולברוח מהאויב שרודף אחריו – זו הדמות האדומה בפינה.  
אם האויב משיג אותך – הפסדת.

**בנוגע לרמות קושי –**  
יש לנו שתי רמות:  
במצב **Normal**, יש אויב אחד, והתנועה קלה יותר.  
במצב **Hard**, יש שני אויבים, וקירות חוסמים נוספים – כלומר מדובר באתגר הרבה יותר רציני.

בנוסף, במהלך המשחק אפשר להיתקל באלמנטים מפתיעים כמו:

* **רנדומייזר**, שמשנה את מבנה המבוך תוך כדי תנועה,
* **תור נוסף**, שמאפשר לשחקן לנוע פעמיים ברצף.

כל אלה הופכים את המשחק לדינמי ומשתנה – והוא אף פעם לא בדיוק אותו הדבר."

שקף 4:  
*"אחד הדברים המרכזיים שבנינו במשחק זה אלגוריתם A-Star, או A.*\*  
זה האלגוריתם שבעצם מפעיל את האויב – כלומר, הוא זה שגורם לו לרדוף אחרי השחקן בצורה חכמה.

A\* הוא שילוב בין Dijkstra (שמחפש את המסלול הכי קצר) לבין חיפוש גרידי (Greedy), שמנסה לנחש לאן הכי משתלם ללכת.

הוא לוקח בחשבון גם את מה שקרה עד עכשיו – מה שנקרא g(n), המרחק שהאויב כבר עבר,  
וגם את ההערכה כמה עוד נשאר – h(n), המרחק המשוער ליעד (שזה השחקן).

ביחד זה נותן את הפונקציה f(n) = g + h – והיא זו שקובעת כל פעם לאן הכי כדאי לאויב לזוז.

היתרון הגדול של A\* זה שהוא גם חכם וגם מבטיח פתרון אופטימלי – כל עוד התנאים במפה לא משתנים יותר מדי.

במשחק שלנו, האלגוריתם הזה פועל בזמן אמת – בכל תור של האויב – והוא עוזר לו לחשוב ולבחור את הצעד הכי טוב כדי לתפוס את השחקן."

שקף 5:  
**"בשקופית הזו אנחנו מסבירים איך האלגוריתם שוקל לאן לזוז.**  
יש פה שני מרכיבים עיקריים:

🔹 **פונקציית הערכה (Heuristic):**  
בודקת כמה רחוק השחקן נמצא מהאויב – זו הדרך של האויב 'לשאוף' להגיע אליו.

🔹 **עלויות משבצות (g):**

* רגילה שווה 1.
* משבצת עם תור נוסף זולה יותר.
* רנדומייזר – מייקר את הדרך.
* וגם קיר סמוך מוסיף עלות קטנה.

בנוסף, האויב מתאים את ההחלטות לפי ההקשר:

* אם הוא קרוב לשחקן – הוא מנסה לרדוף.
* אם הוא קרוב ליציאה – הוא מנסה לחסום ולמנוע בריחה.

ובסופו של דבר –  
🔸 הוא בוחר מסלולים שנראים ממש מתוחכמים,  
🔸 וההתנהגות שלו מרגישה הרבה יותר חכמה – כאילו זה שחקן אמיתי שמגיב אליך."  
  
  
שקף 6:  
**"אז איך בעצם בנינו את ההתנהגות של האויב?**  
האויב במשחק לא רודף בצורה עיוורת, אלא מתאים את עצמו למרחק מהשחקן – והוא מתנהג שונה במצבים שונים.

פיתחנו **ארבעה מצבים אדפטיביים**, לפי המרחק:

1. **מרחק גדול – מצב של רדיפה ישירה:**  
   האויב פשוט מתקדם בכל הכיוונים האפשריים כדי לסגור על השחקן.
2. **מרחק בינוני – מצב של חסימה:**  
   הוא מנסה לחסום את דרכי הבריחה של השחקן, ולאו דווקא לרוץ אחריו.
3. **מרחק קטן – הוא מתחיל לשחק איתך:**  
   מחפש נתיב עקיף או מבלבל, כדי להפתיע אותך – דומה למה ששחקן אמיתי היה עושה.
4. **מרחק אפסי – צמוד לשחקן:**  
   כאן הוא מפעיל חיזוי – מנסה לנחש לאן אתה תלך, ולחתוך אותך שם.

הדבר הכי חשוב הוא שכל החלטה מתבצעת **בזמן אמת** – לפי איפה שהשחקן נמצא באותו רגע.  
אין כאן Machine Learning, אבל כן יש סוג של "למידה" על בסיס תנועה ותגובה – שזה גורם לאויב להיראות הרבה יותר חכם."  
  
שקף 7:  
**"המשחק שלנו נבנה כולו בשפת JavaScript, ויש לזה הרבה יתרונות.**  
קודם כל, זה אומר שהכול רץ בצד הלקוח – בדפדפן – בלי התקנות ובלי הגדרות.

הנה כמה טכנולוגיות עיקריות שהשתמשנו בהן:

1. **Canvas API** – זה הכלי שאיתו ציירנו את כל המבוך והאלמנטים הגרפיים. הוא מאפשר לנו שליטה מלאה על הפיקסלים, ומושלם למשחקים.
2. **DOM ואירועים** – השתמשנו ב־DOM כדי להציג מידע, ולחבר כפתורים לאירועים במשחק. לדוגמה, כשלוחצים על 'Start Game' או 'Tutorial' – נשלפת התצוגה הנכונה.
3. **התאמות למובייל** – הקפדנו על כך שהמשחק יעבוד גם במסכים קטנים, ושלא יהיו תלות בגודל מסך מסוים או רזולוציה.
4. **Local Storage** – זה איפשר לנו לשמור נתוני משחק כמו תוצאה אחרונה או רמת קושי, גם אחרי ריענון של העמוד.

בסופו של דבר, בזכות JavaScript הצלחנו לבנות משחק שהוא גם אינטראקטיבי, גם גרפי וגם נגיש – וכל זה דרך הדפדפן בלבד."

שקף 8  
**"כמו בכל פרויקט – גם כאן היו לא מעט אתגרים בדרך.**  
האתגרים העיקריים לא היו בעיצוב הגרפי, אלא דווקא בלוגיקה של המבוך וההתנהגות של האויב – במיוחד כשמדובר במבוכים רנדומליים שיכולים להיווצר בכל מיני צורות.

🔹 **אתגר ראשון – תקיעות במבוך:**  
לפעמים נוצר מצב שיש אזורים שאי אפשר לצאת מהם – כלומר המבוך נבנה לא תקין.  
אז מה עשינו?  
השתמשנו באלגוריתם BFS כדי לבדוק אם בכלל קיימת דרך מהשחקן ליציאה.  
אם אין – אנחנו בונים את המבוך מחדש, באופן אוטומטי.

🔹 **אתגר שני – אויב טיפש מדי או חכם מדי:**  
היינו צריכים לאזן את ההתנהגות שלו – שלא יהיה קל מדי, אבל גם לא בלתי מנוצח.

הפתרון שלנו היה לפתח **מערכת אסטרטגית דינמית** –  
בהתאם למרחק מהשחקן, האויב משנה את ההתנהגות שלו: רודף, חוסם, צופה תנועה וכו'.  
וככה יצרנו אשליה של אויב שחושב, אבל גם אפשר לנצח אותו – אם מתכננים נכון.

בקיצור, זה דרש הרבה ניסוי וטעייה – אבל זה בדיוק מה שעשה את המשחק מעניין יותר."

שקף 9:  
**"כאן אנחנו מציגים את השינויים שביצענו לעומת ההצעה המקורית.**  
במהלך הפיתוח, ראינו שיש מקום להרחיב ולשדרג את הרעיונות – וזה מה שעשינו:

🔹 **רמת קושי חדשה – Hard mode:**

* הוספנו שני אויבים במקום אחד.
* המבוך נעשה צפוף יותר – עם 35% קירות, במקום 25%.  
  זה הפך את המשחק להרבה יותר מאתגר, ודורש תכנון הרבה יותר מדויק.

🔹 **שדרוג מנגנון הניקוד:**  
הוא כבר לא מבוסס רק על ההגעה ליציאה, אלא גם על איסוף מטבעות לאורך הדרך – מה שדוחף את השחקן להסתכן ולחפש יותר.

🔹 **שיפור הבינה המלאכותית:**  
במקום רדיפה פשוטה, הוספנו את ארבעת מצבי המרחק שראינו קודם – מה שהפך את האויב להרבה יותר מגוון וחכם.

🔹 **Power-ups חדשים:**

* תור נוסף – שמאפשר לשחקן לנוע פעמיים ברצף.
* רנדומייזר – משנה את המבוך בזמן אמת.

ובסוף – **שיקלול כללי**: השינויים האלה הפכו את המשחק לדינמי, מעניין ומאתגר יותר – בלי לשבור את האיזון או לפגוע בחוויית המשחק."  
  
  
שקף 10:  
**"אז נסיים בהצגה של הארכיטקטורה הכללית של הפרויקט.**  
MazeRun בנוי בצורה מאוד קלה ויעילה – בלי שימוש במסגרות חיצוניות, ובלי תלות בשרת.

🔹 **מבנה תיקיות מסודר:**

* קבצי **HTML** משמשים לתפריטים, המשחק עצמו, וההדרכה.
* קבצי **JavaScript** מופרדים לפי תחום – יש קובץ ללוגיקה של המשחק, אחד להדרכה, ואחד למסך הראשי.
* קובץ **CSS** אחד אחראי לעיצוב רספונסיבי שמתאים לכל הרזולוציות – גם במובייל.

🔹 **שימוש בטכנולוגיות פשוטות אבל חזקות:**

* עם **Canvas API** יצרנו גרפיקה דו-ממדית בזמן אמת.
* **JavaScript** שימשה אותנו להכל – הלוגיקה, התנועה, והבינה המלאכותית.
* השתמשנו גם ב־**SweetAlert2** להצגת הודעות קופצות בצורה נוחה ואסתטית.

🔹 בנוסף, הקפדנו שהמערכת תהיה **קלה, מהירה, ולא תלויה בשרת** – הכול פועל מהדפדפן בלבד, ואפילו לא חייב חיבור לאינטרנט.

לסיכום, הארכיטקטורה פשוטה – אבל כל רכיב מופרד, ברור, ומאפשר תחזוקה ושדרוג בהמשך בצורה קלה."  
  
שקף 11  
🔹 **פונקציות הערכה חכמות:**  
האלגוריתם יודע להתאים את עצמו לסיטואציה – למשל, הוא מחשב בונוס או עונש לפי אם השחקן קרוב ליציאה או לא.

🔹 **אופטימיזציות בביצועים:**

* המסלולים של האויב מתעדכנים רק כשהוא בתור – כדי לחסוך חישובים מיותרים.
* וכשמייצרים את המבוך, אנחנו משתמשים ב־BFS כדי לבדוק מראש שהוא פתיר.

🔹 **חוויה מאוזנת ומאתגרת:**

* יש שתי רמות קושי לבחירה,
* ויש גם אלמנטים שמוסיפים עניין – כמו מטבעות, תור נוסף ורנדומייזרים.
* והכול משתלב יחד כדי לתת חוויית משחק כיפית, אבל לא קלה מדי.

🔹 **ממשק מודרני ונוח:**

* כל ההודעות מוצגות עם SweetAlert2,
* והעיצוב מותאם לתפקידים – בין אם שחקן או מדריך.

ולבסוף –  
🔸 **המשחק מתאים לדפדפן**, קליל, מהיר, ולא דורש התקנה בכלל. פשוט לוחצים ומשחקים.

ככה שהצלחנו לבנות משהו שגם עובד טוב טכנית – וגם כיף לשחק בו."

שקף 12  
🔹 **עמוד הבית** – מאפשר להתחיל משחק חדש או להיכנס למדריך.  
כאן גם מופיעים הפיצ'רים העיקריים של המשחק – כמו איסוף מטבעות, רמות קושי, Power-ups, ועוד.

🔹 **עמוד המשחק** – כאן הכול קורה.  
השחקן נע במבוך אקראי, צריך לאסוף מטבעות ולהגיע ליציאה – וכל זה תוך כדי שהוא נמנע מהאויב.  
במצב Hard יש שני אויבים והרבה יותר קירות, כך שהאתגר גובר.  
למעלה רואים את התוצאה, מצב המשחק, ואת המטרה הנוכחית.

🔹 **עמוד ההדרכה** – מדריך פשוט שמסביר איך לשחק, עם שלבים מדורגים ודמויות שממחישות מה מותר ואסור לעשות.

בסופו של דבר – רצינו לבנות משחק שהוא גם מאתגר, גם אינטואיטיבי, וגם נעים לעין – והצילומים האלה ממחישים את זה בצורה הכי טובה."

שקף 13:  
**"נסיים בכמה מילים על ההישגים שלנו בפרויקט.**

✅ קודם כול, המשחק עובד באופן מלא – גם בזמן אמת, גם עם שליטה מלאה, והוא נבנה בצורה יציבה.  
✅ שילבנו בתוכו בינה מלאכותית שמתאימה את עצמה לסיטואציה, מה שהפך את ההתנהגות של האויב לחכמה ומגוונת.  
✅ דאגנו לכך שכל מבוך שיווצר יהיה תמיד פתיר – בעזרת בדיקת תקפות עם BFS עוד לפני שמתחילים לשחק.  
✅ חוויית המשחק כוללת רמות קושי, ניקוד, אלמנטים ייחודיים כמו רנדומייזרים ותור נוסף – וכל זה תורם להרגשה של משחק עשיר ומעניין.  
✅ כל הקוד נבנה בצורה מודולרית – כך שנוכל להוסיף או לשנות דברים בקלות בהמשך.  
✅ ובנוסף – שילבנו אופטימיזציות, עיצוב רספונסיבי, ונגישות מלאה – בלי תלות בשרת או התקנה.

בסופו של דבר, אנחנו גאים בתוצאה – גם מבחינת חוויית המשחק, וגם מבחינת היישום הטכני."