**<TRIVR>**

מסמך אפיון

<יותם לויט>

<גרסה X>

<תאריך>

**היסטוריית גרסאות המסמך**

|  |  |  |
| --- | --- | --- |
| **תאריך** | גרסה | **תקציר השינויים** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |



1. **הקדמה**

1.1 מטרה

*מטרתו של מסמך זה, היא להציג את הפרויקט בצורה מפושטת ואת חלקיו. המסמך הזה מיועד לכל מי שרוצה להבין לעומק את הפרויקט ואיך הוא עובד בצורה פשוטה ככל שניתן בלי להיכנס לקוד.*

* 1. המוצר
* **סימולטור TRIVER**
* *שם המוצר הוא* TRIVER – Test Robot In Virtual Reality . TRIVER *הוא סימולטור בדיקה אשר מטרתו להקנות סביבה וירטואלית, שבה ניתן לבחון רובוטים ללא חשש לפגיעה פיזית בהם או בסביבה*. *יתרון TRIVER על פני בדיקה של הרובוט בעולם הפיזי היא שעל ידי בדיקה בסימולטור בעלי הרובוט יוכלו להימנע מתאונה מסוכנת בעולם הפיזי.*
  1. הגדרות

**לסמלץ – להפעיל סימולטור**

**לקוחclient – משתמש המסמלץ את הרובוט שלו בסימולטור**

**שרת – מחשב מרוחק בו TRIVER מורץ** .

1. תיאור כללי

החלק הזה מתאר את הגורמים העיקריים המשפיעים על המוצר ועל דרישותיו. חלק הזה לא מפרט דרישות ספציפיות אלא רק עוזר להבין את הדרישות.

2.1 פונקציונליות

1. בדיקה ווירטואלית

**הרובוט ייבדק בסביבה ווירטואלית במחשב**

1. שינוי סביבת הבדיקה

**ללקוח תהינה האפשרות לשנות את סביבת הבדיקה**

1. תצוגה גרפית

**TRIVER יראה את הסימולטור בתצוגה גרפית**

1. שליטה ידנית או לפי אלגוריתם ממשתמש מרוחק

**ללקוח תהיה אפשרות בחירה בהפעלת הרובוט הסימולטור – ידנית או לפי אלגוריתם**

1. הרצה של כמה רובוטים באותה סביבה

**הרצה של מפר רובוטים בו זמנית בסביבת עבודה אחת**

2.2 קהל היעד

**קהל היעד של המערכת – אנשים, חברות, קבוצות רובוטיקה אשר רוצים לבחון את הרובוט שהם בנו בסביבה ווירטואלית ולמנוע פגיעה פיזית שהרובוט יוכל לגרות הסביבה הפיזית.**

2.3 אילוצים עיקריים

- **המערכת צריכה לעשות מספר תהליכים בו זמנית**

-  **המערכת דורשת חיבור לאינטרנט בכדי לשנע מידע בין הלקוח לשרת**

2.4 הנחות ותלויות

**הסימולטור ירוץ בסביבת ווינדוס.**

1. דרישות מפורטות

החלק העיקרי והחשוב ביותר של מסמך האפיון. פרק זה יכיל את הדרישות המפורטות מהמערכת שינחו את מעצבי המערכת, המפתחים, והבודקים בהמשך. כל דרישה שתפורט בחלק זה צריכה להיות:

* מזוהה על ידי שם (או עדיף - מספר) באופן ייחודי
* מתועדפת (כלומר מה העדיפות שלה כלפי הדרישות האחרות)
* ניתנת לבדיקה
* נכונה ושלמה
* עקבית ולא מתנגשת עם דרישות אחרות

*שימו לב שישנם מספר סוגים של דרישות, כפי שמפורט בכותרות כאן:*

*יש להשקיע בחלק זה ולפרט את הדרישות בצורה ברורה ונוחה להבנה. יש לזכור שזה לא מסמך עיצוב- יש להמנע מתיאור מפורט מדי שמגביל את העיצוב הטכני (למשל – שפת תכנות) שיגיע בשלב מאוחר יותר של עבודתכם.*

*את הדרישות רצוי לחלק לקטגוריות הבאות:*

3.1 דרישות פונקציונליות

*תיאור של פיצ'רים ספציפיים במערכת. לכל דרישה פונקציונלית מומלץ לפרט:*

* *הסבר כללי*
* *קלט*
* *תהליך*
* *פלט*
* *טיפול בשגיאות*

*ניתן (ואף רצוי) להציג את הדרישות הפונקציונליות בפורמט של* [*תרחיש שימוש (use case)*](http://he.wikipedia.org/wiki/%D7%AA%D7%A8%D7%97%D7%99%D7%A9_%D7%A9%D7%99%D7%9E%D7%95%D7%A9)*. בכל "נקודת זמן" של תיאור התרחיש יש לפרט את כל האפשרויות השונות שעומדות בפני המשתמש או המערכת, ומה יכול לקרות בכל התפצלות שכזו (שימו לב להתייחס גם למקרים שליליים, כמו שגיאות או נתונים לא מתאימים, ולא להניח שהמשתמש תמיד פועל בדרך שלה אנו מצפים)*

3.2 דרישות של ממשקים חיצוניים

**3.2.1 ממשקי משתמש** (למשל: GUI)

**3.2.2 ממשקי חומרה**

**3.2.3 ממשקי תוכנה** (אם התכנה מחולקת לכמה תוכנות שונות שצריכות לתקשר בניהן)

**3.2.4 ממשקי תקשורת (למשל תקשורת בין שר**ת ללקוחות)

3.3 דרישות לא פונקציונליות

*דרישות מהמערכת שלא מתבטאות בפיצ'ר ספציפי או בתהליך ספציפי שמתרחש במערכת אבל משפיעות על אופן עיצובה ומימושה, לדוגמא:*

**3.3.1 דרישות ביצועים (performance)**

**3.3.2 דרישות מהימנות (reliability)**

**3.3.3 דרישות זמינות (Availability)**

**3.3.4 דרישות אבטחה (security)**

**3.3.5 דרישות תחזוקה (maintainability)**

**3.3.6 דרישות ניידות (portability)**

3.4 דרישות בסיס נתונים

* *איזה נתונים יישמרו*
* *באיזה פורמט יש לשמור את המידע (DB / קובץ . ניתן לציין גם כמה אופציות)*
* *מה יכולות האחסון הנדרשות (נפח אחסון לדוגמא)*
* *מה רמת האבטחה הנדרשת*

3.5 דרישות נוספות

*כאלה שלא ידעתם לאיזה חלק הם שייכות*

5. נספחים