

**בית-הספר להנדסה**

**המחלקה להנדסת חשמל ומחשבים**

**התוכנית להנדסת מחשבים**

**ספר פרויקט גמר לשנת תשפ"א**

**נושא הפרויקט:**

**IIOT Data Mining**

המבצעים: שקד אברהם ,שחר כדורי .

המנחים: מר' תמיר דרשר המרכז האקדמי רופין .



**בית-הספר להנדסה**

**המחלקה להנדסת חשמל ומחשבים**

**התוכנית להנדסת מחשבים**

**ספר פרויקט גמר לשנת תשפ"א**

**נושא הפרויקט:**

**IIOT Data Mining**

המבצעים: שקד אברהם ,שחר כדורי .

המנחים: מר' תמיר דרשר המרכז האקדמי רופין .

**הצהרה: העבודה המתוארת במסמך זה היא תוצאה של מחקר אישי שלי. כל טקסט או תוצאה שנלקחו והוכנסו לעבודה זו ממקורות אחרים מתועדים ככאלה. אני יודע שאי עמידה בתנאים (של עבודה עצמית, וציטוט נאות של מקורות) היא עבירה על תקנון המשמעת של בית הספר העשויה לגרור צעדים משמעתיים בפני ועדת המשמעת.**

חתימת הסטודנט..................... תאריך.......................

חתימת המנחה..................... תאריך.......................

חתימת המנחה..................... תאריך.......................

**תקציר**

מטרת המערכת היא להפוך את המידע המתקבל ממכונות יצור וציודי בדיקה אשר מפיקים לוגים לידע כלומר להפוך את המידע לקריא ונגיש ובכך להוות בסיס עבור ניהול וכריית מידע. כריית המידע היא איסוף דוחות על מכונות המכילות מידע על תפוקתן.

המערכת מאפשרת הגדרת תהליכי איסוף של המידע מהמכונות, ביצוע פעולות עיבוד על המידע לצורך קבלת תצורה אחידה ושמירתו לצורך חקר בהמשך.

למעשה המערכת שלנו משלבת את כלל הדוחות המתקבלים מן המכונות ומבצעת עליהם טרנספורמציה כך שיתאימו לפורמט בעל יכולת התממשקות לשרת חיצוני.

אופן קבלת הלוגים, סידורם ושמירתם מתבצע בזמן קבוע, אחת ליום ובכך יסופק מידע מהימן אודות מצב המכונות ופעולתן.

תפקידה העיקרי של המערכת הינה שאיבת כל הלוגים המתקבלים ממכונות הבדיקה השונות בפורמטים שונים, ביצוע טרנספורמציה עליהם והצגת כל הדו"חות הללו למשתמשים.

בניית מערכת הכוללת 2 תתי מערכות:

מערכת ניתוח דפי HTML המבצעת שליפה, תאימות, מעבר ותיזמון.

מערכת תצוגה המבצעת בניתוח והצגה.

תחילה המערכת מקבלת קובץ קונפיגורציה (קובץ JSON) המכיל עבור אלו תגי HTML יש לתרגם את המידע. המערכת הראשונה מבצעת שליפה של הלוגים ממסד הנתונים אליו מחוברות המכונות, הלוגים המתקבלים ממסד הנתונים הם בפורמט HTML וכל דוח שונה בצורתו מהשני. לאחר מכן המערכת מבצעת עליהם טרנספורמציה בעזרת ספרייה HTML Agility שתפקידה לזהות תגי HTML ועבור כל תג מציגה את כל התכונות שלו במחלקה מיועדת. המערכת תבצע הכנסה לDataTable (שמירה לוקאלית במערכת). ברגע שפועלה זו בוצעה, המערכת תשמור את המידע בפורמט (Comma-Separated Values)CSV בתיקיה מקומית על פי קובץ ההגדרה. לאחר מכן המערכת תעביר את המידע שהתקבל למערכת התצוגה. מערכת זו תקבל את קובץ הCSV ותאפשר הצגה של הדוחות למשתמשים בצורה נגישה ומסודרת על גבי מערכת הנקראת . ELK (ELastic search and Kibana)

זוהי מערכת חינמית הכוללת בתוכה מקבץ פרויקטים של קוד פתוח המאפשרים ניתוח והצגת לוגים על גבי פאנלים ויזואליים. היא מאפשרת ניתוח וחיפוש של המידע הקיים בפורמטים שונים: JSON (java script object notation) ו (Comma-Separated Values)CSV.

פרויקט זה התמקד בפיתוח מערכת העובדת כתהליך רקע ומספקת למשתמש ניתוח מפורט ומהימן אודות פעילות המכונות באופן שוטף וללא תלות בסוג המכונה / המכונות במפעלים או הלוגים השונים שמיוצרים על ידם, וזאת על מנת לספק פתרון למפעלים עם מכונות ישנות לבדיקת ציוד המייצרות לוגים בפורמטים שונים ולא אחידים. המערכת מספקת תהליך נכון של שליפה ותאימות, המרת המידע ,העברתן בצורה טובה וניתוח ותצוגה בצורה קריאה ומובנת למשתמש.

**תוכן עניינים**

**1. הקדמה5**

**2. סקר ספרות5**

2.1 הדוחות המתקבלים מן המכונות5

2.2 הגדרת הדוח6

2.3 מערכת ניתוח דפי HTML6

2.4 מערכת תצוגה6

2.5 חיבור המערכות6

2.6 תרשים המערכת7

**3. תכנון הנדסי8**

3.1 סביבת הפיתוח8

3.2 ארכיטקטורת המערכת8

3.3 דרישות המערכת9

3.4 Use Case Diagram10

3.5 טבלה המתארת את ה- Use Case Diagram11

3.6 Use Case Glossary 12

3.7 Ttraceability Matrix15

3.8 Sequence Diagram 16

3.9 Class Diagram 18

3.10 המחשת המערכת 19

**4. תוצאות הבדיקה והדגמות24**

**5. מסקנות והמלצות26**

**6. רשימת מקורות27**

**7. Abstract28**

**8. נספחים30**

**1. הקדמה**

IIOT Data Mining (Industrial Internet of Things)- מורכב משלושה חלקים:

1. Internet of things – רשת מכשירים אלקטרונים המחוברים לאינטרנט.
2. Industrial -מפעלים בתוכם קיימת רשת של מכונות. אצלנו בפרויקט מדובר במכונות לבדיקת ציוד, המחוברות ברשת משותפת.
3. Data Mining - כריית נתונים, הידוע גם בשם גילוי ידע בנתונים (KDD). כריית נתונים הוא תהליך של חשיפת דפוסים ומידע יקר אחר ממערכות נתונים גדולות.

פרויקט זה עוסק בכרייה של מידע המגיע מלוגים המיוצרים על ידי מכונות ישנות המחוברות לרשת של המפעל, מכונות אלו מבצעות בדיקות לציוד אלקטרוני כגון: קסדות תרמיות, וולטמטר וכו'. על המידע שנכרה להיות מוצג באופן ברור למשתמש במערכת. משתמש במערכת מוגדר להיות אדם או תהליך שצריכים לעשות שימוש במידע המגיע מהמכונות, למשל: מנהל מפעל המעוניין לראות פרטים על המכונות והציוד במפעל בו הוא עובד על מנת לדעת כיצד יוכל לשפר את תהליך הייצור.

במהלך הפעלת המכונה, המכונה מייצרת דו"ח על ביצוע פעולתה. בסיום, כל מכונה מעבירה את הלוגים אל המסד נתונים.

2**. סקר ספרות**

2.1 **הדוחות המתקבלים מהמכונות**

הדוחות שהמכונות מפיקות מכילים מידע אודות פעולתן של המכונות לאחר ביצוע בדיקתן על הציוד. דו"חות אלו נקראים בתעשייה לוגים. הדו"חות יכולים להיכתב במספר רב של פורמטים. אחד הפורמטים הנפוצים הוא HTML - .hyper text markup language דפדפני אינטרנט הם כלי נפוץ ולכן כתיבת דוחות בפורמט אשר יתממשק עם דפדפני אינטרנט יהווה יתרון בהעברת המידע ולכן בפרויקט זה נבחר לעשות שימוש בפורמט HTML.

HTML היא שפה שמאפשרת הצגת מידע ברשת האינטרנט. היא מאפשרת לבנות את השלד של העמוד ואומרת לדפדפן מה יש להציג למשתמש. היא שפת סימון המורכבת מאוסף של תגיות המכתיבות לדפדפן את הצורה בה עליו להציג את הדפים. אוסף התגיות המעצבות את הדף נקראות קוד הדף או קוד המקור. תגיות HTML נכתבות בין הסימנים <>. היא משמשת להגדרת מבנה התוכן.

עיצוב העמוד מוגדר באמצעות קוד CSS - cascading style sheets. היא שפה שמגדירה את תבנית העיצוב של דפי הHTML. מטפלת בגופנים, צבעים, גבולות, שורות וכו. משמשת לעיצוב תוכן מובנה.

דוגמה לדוח שמופק על ידי המכונות בפורמט HTML:

A picture containing application

Description automatically generated

2.2 **הגדרת הדוח**

דוח המתקבל מהמכונות יכול להיות: תקין או לא תקין.

**דוח תקין** הוא דוח שלם המכיל את כל המידע שהמכונה אספה בזמן הפעלתה. ע"י שליפה והמרה ניתן יהיה להפיק ממנו מידע.

**לוג לא תקין** הוא דוח לא שלם שבזמן הפעלת המכונה ואיסוף המידע התגלתה תקלה כלשהי שגרמה לכשל במכונה. כשל זה מונע את השלמה הדוח ולכן לא מתאפשרת שליפה והמרה של הדוח. לדוגמא הפסקת חשמל.

2.3 **מערכת ניתוח דפי HTML**

במערכת ישנו קובץ הגדרות הכתוב בפורמט JSON-Object Notation JavaScript אותו המערכת מקבלת ופועלת על פיה. בחירה בפורמט זה היא מכיוון שקבצי JSON נחשבים לפורמט נפוץ בתעשייה בזכות הפשטות והגמישות שהיא מאפשרת.

JSON הוא פורמט טקסטואלי, הקריא לאדם המיועד להעברת מבני מידע. הוא מבנה קל משקל להחלפת מידע וקל לפיענוח ע"י מכונות. הוא מבוסס על תת מערכת של שפת התכנות JavaScript.

בנוי על שני מבנים:

* אוסף של זוגות בעלי שם/ערך. בשפות השונות, זה בא לידי מימוש כ-אובייקט, רשומה (Record), מבנה (struct), מילון (‎dictionary‏), Hash-table, ‎Keyed list‏ או מערך אסוציאטיבי (Associative array).
* רשימה מסודרת של ערכים. ברוב השפות, זה בא למימוש כמערך (‎Array‏), vector, רשימה (list) או sequence.

אלו הם מבני מידע אוניברסליים. למעשה כל שפות התכנות המודרניות תומכות בהם בצורה זו או אחרת.

המערכת שולפת את ההגדרות מקובץ הקונפיגורציה ואת הדוחות בפורמט HTML ממסד הנתונים. תחילה, המערכת תנתח את הדוחות ע"י שימוש בקובץ ההגדרות. לאחר מכן, המידע יישמר בטבלאות מקומיות ובסיום התהליך יתקבל קבצי CSV המכילים את הטבלאות.

המערכת פותחה בשפת c# ב.Visual Studio

2.3 **מערכת תצוגה**

מערכת התצוגה בוצעה ע"י שימוש ב ELK-.Elastic search and Kibana זוהי מערכת לאיסוף נתונים וסידורם. כלומר המערכת מסדרת ומאפיינת את המידע כך שיהיה קל יותר לחפש, לבדוק ולאתר לוגים מתוך מערכת מסוימת. המערכת עוזרת לנתח, לאגד ולאסוף נתונים ממערכות שונות.

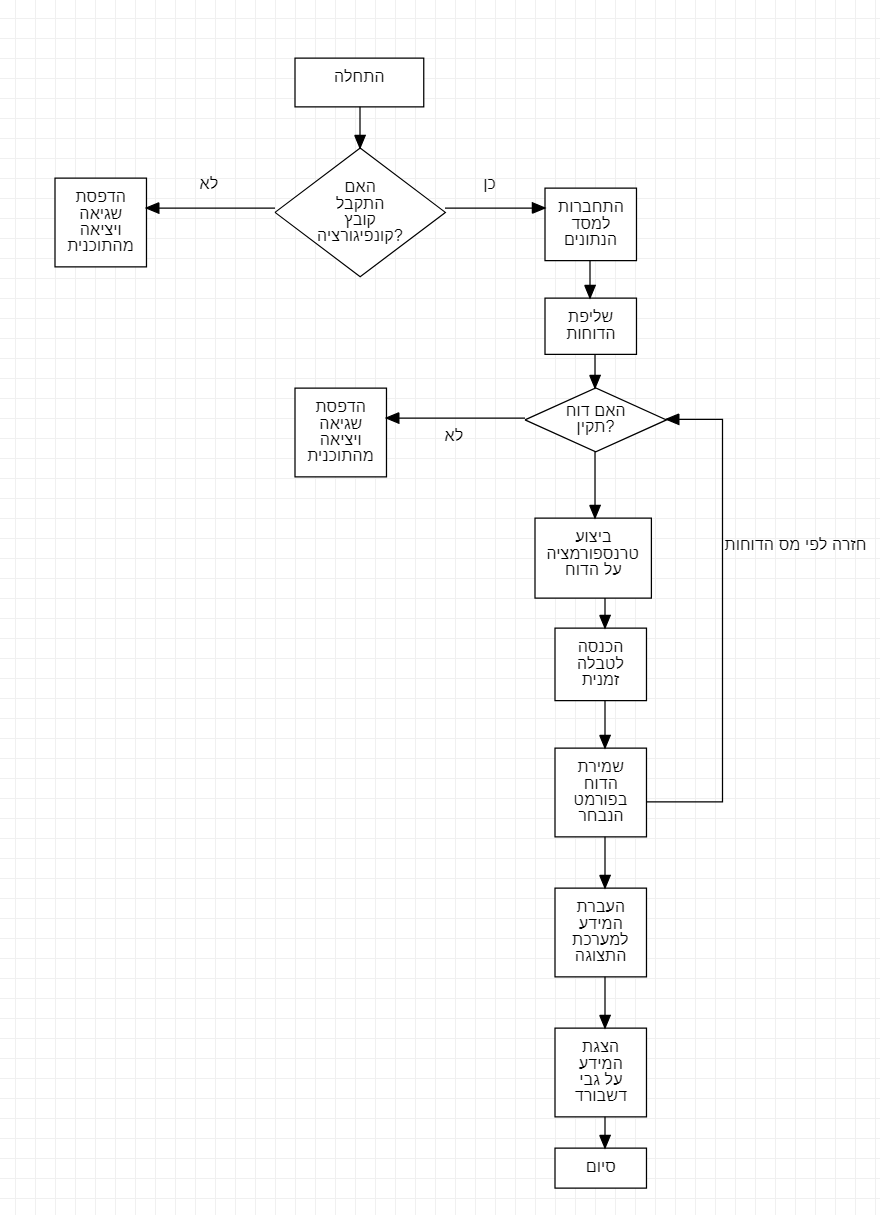
הכלים שבונים את ELK הם:

* Elasticsearch -מערכת לניתוח טקסט מבוססת Apache
* Logstash- מערכת לוגים האוספת ומאגדת אותם. אוספת לוגים ממערכות שונות.
* Kibana- מערכת הוויזואלית, המציגה בצורה נוחה למשתמש את הלוגים, ומכילה את בתוכה את Elasticsearch ו Kibana ושלל אפליקציות העוזרות לניתוח המערכת.

2.4 **חיבור המערכות**

העברת המידע ממערכת ניתוח דפי HTML אל מערכת התצוגה מתבצע ע"י טרנספורמציה לפורמט קריא. בפרויקט הוחלט לעשות שימוש בפורמט CSV מכיוון שזהו פורמט קל לכתיבה בו ניתן לעשות שימוש בשתי המערכות. CSV – comma separated values הוא סוג מיוחד של קובץ שניתן ליצור או לערוך בExcel. קבצי CSV מאחסנים מידע המופרד באמצעות פסיקים. קל להעביר אותם מתוכנית אחת לאחרת.

2.5 **תרשים המערכת**



איור 1 תרשים המערכת

3. **תכנון הנדסי**

3.1 **סביבת פיתוח**

* סביבת פיתוח Visual Studio - לכתיבת הקוד של האפליקציה.
* סביבת פיתוח SQL Express - לכתיבת ה DataBase האפליקציה.
* סביבת פיתוח ELK- להצגת המידע

3.2 **ארכיטקטורת המערכת**

המערכת מקבלת את הפרמטרים מקובץ ההגדרות של הלקוח , מעדכנת את הגדרות המערכת על מנת לבצע המרה בצורה נכונה, מבצעת שליפת לוגים עבור כל מכונה, מבצעת המרה מפורמט HTML לפורמט CSV ,מבצעת העברה של הקבצים המפורמטים אל מערכת ה ELK ובאמצעות שימוש מערכת זו היא מבצעת עיבוד וניתוח על הנתונים כלומר מבצעת חיפוש וניתוח. לאחר מכן היא מציגה את המידע באמצעות גרפים, זה יבוצע על גבי Dashboards באמצעות מערכת הצגת הנתונים הפועלת יחד עם Elasticsearch הנקראת Kibana.

Diagram

Description automatically generated

איור 2 ארכיטקטורת המערכת

3.3 **דרישות המערכת**

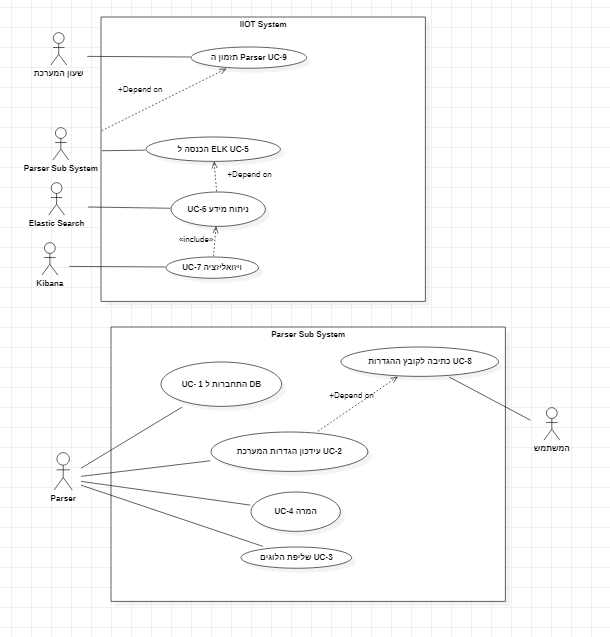
תיאור דרישות המערכת לפי עדיפויות על מנת לספק תכנון ואפיון נכון, מדויק ויעיל לפיתוח הפרויקט.

|  |  |  |
| --- | --- | --- |
| Requirement | Priority | Identifier |
| המכונות יעלו את הדו"חות באופן עצמאי על מנת שהמידע יהיה נגיש לשליפה מבסיס הנתונים הקיים. | 2 | REQ01 |
| המערכת צריכה לספק אפשרות למשתמש להגדיר אלמנטים במבני לוגים שונים ל parsing ומה לפרסר על מנת לבצע parsing בצורה דינאמית ע"י קובץ הגדרות. | 1 | REQ02 |
| המערכת צריכה לקרא את קובץ ההגדרות כל יום בהתאם לזמני ביצוע ה parsing. | 2 | REQ03 |
| המידע מהלוגים צריך להיות נגיש לקריאה עד יום מרגע הכנסתם לבסיס הנתונים. | 3 | REQ04 |
| המידע מהלוגים צריך להיות מומר ומועבר ל ELK מידי יום באופן שוטף 24/7. | 2 | 5REQ0 |
| המערכת צריכה להציג את הנתונים על גבי ממשק נוח וקריא אשר יספק מידע עדכני ובזמן אמת 24/7 אשר יהיה זמין למשתמש. | 4 | 6REQ0 |
| המשתמש אחראי לשנות את תוכן קובץ ההגדרות על מנת להתחיל את תהליך הפעלת המערכת. | 5 | 07REQ |
| לוגים לא תקינים שיתקבלו על ידי המערכת לא יעברו המרה והמידע שלהם לא יעבור הלאה ל Elastic Search. | 4 | REQ08 |
| המידע המומר שמגיע מהלוגים צריך להיות מנותח בצורה כלשהי לפרטים על מנת לספק מידע מדויק ואמין על גבי מצב המוצרים שהמכונות בודקות. | 1 | REQ09 |

טבלה 1 דרישות המערכת

**Use Case Diagram** 3.4

דיאגרמה התנהגותית אשר מנתחת תרחישי שימוש:



איור 3 UML use case

3.5 **טבלה המתארת את ה-Use Case Diagram**

|  |  |  |
| --- | --- | --- |
| Use Case Name | Actor’s Goal (what the actor intends to accomplish) | Actor |
| התחברות ל DB (1-UC) | יצירת חיבור בין מסד הנתונים הנוכחי למערכת IIOT. | Parser |
| עדכון הגדרות המערכת (UC-2) | מעדכן את ההגדרות המערכת למה שרשום בקובץ ההגדרות בשביל להמיר בצורה נכונה את הלוגים על פי קובץ ההגדרות. | Parser |
| שליפת הלוג (UC-3) | שליפת הלוג הרלוונטי מתוך מסד הנתונים על פי ההגדרות. | Parser |
| המרה (UC-4) | המרת הלוג מדף HTML לפורמט CSV. | Parser |
| הכנסה לELK (UC-5) | ביצוע תקשורת עם Elastic Search לצורך העברת הלוג בפורמט CSV אליו. | Parser |
| ניתוח מידע (UC-6) | חיפוש ואנליזה בזמן אמת של המידע מהלוגים. | Elastic Search |
| ויזואליזציה ( UC-7) | מציגה את המידע על גבי dashboard מאפשרת visualization של המידע. | Kibana |
| כתיבה לקובץ הגדרות (UC-8) | כותב לתוך קובץ ההגדרות את ההגדרות לפיהם הוא מעוניין לפרסר את קבצי הלוג שלו. | המשתמש |
| תיזמון ה Parser. (UC-9) | מתזמנת את הParser לפי זמני המערכת. | שעון המערכת |

טבלה 2 תיאור Use Case Diagram

3.6 **Use Case Glossary**

|  |  |
| --- | --- |
| Use Case UC-1: התחברות למסד הנתונים | |
| Related Requirments | REQ01, REQ04 |
| Initiating Actor | Parser |
| Actor’s Goal | התחברות למסד הנתונים |
| Preconditions | קבלת הפרמטרים של מסד הנתונים מקובץ ההגדרות. |
| Flow of Events for Main Success Scenario | |
| 1. The System established a connection with the DB. 2. The System ready to get the configuration data. | |
| Flow of Events for Main Failure Scenario | |
| 1. The System did not established a connection with the DB.  2. The System is not ready to get the configuration data. | |

טבלה3 התחברות למסד הנתונים

|  |  |
| --- | --- |
| Use Case UC-2: עדכון הגדרות המערכת | |
| Related Requirments | REQ02, REQ03, REQ08 |
| Initiating Actor | Parser |
| Actor’s Goal | מעדכן את ההגדרות המערכת למה שרשום בקובץ ההגדרות בשביל להמיר בצורה נכונה את הלוגים על פי קובץ ההגדרות, זאת בשל מקרה שמישהו הכניס הגדרות אחרות מאלה שהכניסו בהתחלה לתוך קובץ הjson . |
| Preconditions | חיבור עם מסד הנתונים 1-UC, כתיבה לקובץ ההגדרות UC-8, תיזמון הParser UC-9. |
| Flow of Events fo Main Success Scenario | |
| 1. The Parser take the input (json config file) and stores\update the suitable json configuration file object. 2. The Parser read the propreties of the json to decide how to parse 3. The Parser start parsing according to the configurations | |
| Flow of Events for Main Failure Scenario | |
| 1. Configuration file is empty, thus the Parser take empty empty propreties. 2. The Parser behave by its default. | |

טבלה 4 עדכון הגדרות המערכת

|  |  |
| --- | --- |
| Use Case UC-3: שליפת הלוג | |
| Related Requirments | REQ04, REQ08 |
| Initiating Actor | Parser |
| Actor’s Goal | שליפת הלוג הרלוונטי מתוך מסד הנתונים על פי ההגדרות. |
| Preconditions | התחברות למסד הנתונים UC-1, תזמון ה Parser UC-9. |
| Flow of Events for Main Success Scenario | |
| 1. The Parser take as input a valid log. 2. The Parser start the process in which the log is parsed. | |
| Flow of Events for Main Failure Scenario | |
| 1. The Parser take as input an invalid log.  2. The Parser continue to the next log. | |

טבלה 5 שליפת הלוג

|  |  |
| --- | --- |
| Use Case UC-4: המרה | |
| Related Requirments | REQ05 |
| Initiating Actor | Parser |
| Actor’s Goal | המרת הלוג מדף HTML לפורמט CSV. |
| Preconditions | שליפת לוג UC-3, עדכון הגדרות 2-UC, תזמון הParser UC-9. |
| Flow of Events fo Main Success Scenario | |
| 1. The Parser parse the input log into suitable object to store, and from there to a CSV format. | |

טבלה 6 המרה

|  |  |
| --- | --- |
| Use Case UC-5: ELK הכנסה ל | |
| Related Requirments | REQ05 |
| Initiating Actor | Parser |
| Actor’s Goal | ביצוע תקשורת עם Elastic Search לצורך העברת הלוג בפורמט CSV אליו. |
| Preconditions | המרת הלוג UC-4, תזמון ה Parser UC-9 |
| Flow of Events fo Main Success Scenario | |
| 1. The Parser succefully inserts the CSV files to ELK 2. The ELK get the data process it and analyze with the help of the tool called elastic search. | |

טבלה 7 הכנסה לELK

|  |  |
| --- | --- |
| Use Case UC-6: ניתוח מידע | |
| Related Requirments | REQ09 |
| Initiating Actor | ElasticSearch |
| Actor’s Goal | עיבוד הנתונים כלומר מבצעת חיפוש וניתוח (אנליזה) של הנתונים שהתקבלו כקבצי CSV |
| Preconditions | העברת קבצי CSV ל elastic search UC-5. |
| Flow of Events fo Main Success Scenario | |
| 1. The Parser succefully inserts the CSV files to ELK 2. The ELK get the data process it and analyze. | |

טבלה 8 ניתוח המידע

|  |  |
| --- | --- |
| Use Case UC-7: Visualizing | |
| Related Requirments | REQ06 |
| Initiating Actor | KIBANA |
| Actor’s Goal | מציגה את המידע על גבי dashboard מאפשרת ויזואליזציה של המידע. |
| Preconditions | ניתוח המידע UC-4 |
| Flow of Events fo Main Success Scenario | |
| 1. The Kibana get the analyzed data from ElasticSearch. 2. Kibana visualize the analyzed data in different ways. | |

טבלה 9 ויזואליזציה

|  |  |
| --- | --- |
| Use Case UC-8: כתיבה לקובץ ההגדרות | |
| Related Requirments | REQ02, REQ07 |
| Initiating Actor | משתמש המערכת |
| Actor’s Goal | כתיבה לתוך קובץ ההגדרות את ההגדרות לפיהם המשתמש מעוניין לפרסר את קבצי הלוג שלו. |
| Preconditions | None |
| Flow of Events fo Main Success Scenario | |
| 1. The system user write the configurations he wants in the json configuration file that comes with the system. | |

|  |  |
| --- | --- |
| Use Case UC-9: Parser תזמון ה | |
| Related Requirments | REQ03, REQ05, REQ06, REQ04 |
| Initiating Actor | שעון המערכת |
| Actor’s Goal | מתזמנת את ה Parser לפי זמני המערכת שהוגדרו בדרישות שמצריכות תזמון. |
| Preconditions | None |
| Flow of Events fo Main Success Scenario | |
| 1. The system clock scheduling the Parser. | |

טבלה 10 כתיבה לקובץ ההגדרות

טבלה 11 תזמון ההמרה

3.7 **Traceability Matrix**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| REQS | PW | | UC01 | | UC02 | | UC03 | | UC04 | | UC05 | | UC06 | | UC07 | | UC08 | | UC09 | |
| REQ01 | 2 | | X | |  | |  | |  | |  | |  | |  | |  | |  | |
| REQ02 | 2 | |  | | X | |  | |  | |  | |  | |  | | X | |  | |
| REQ03 | 1 | |  | |  | |  | |  | |  | |  | |  | |  | | X | |
| REQ04 | 3 | | X | | X | | X | |  | |  | |  | |  | |  | | X | |
| REQ05 | 2 | |  | |  | |  | | X | | X | |  | |  | |  | | X | |
| REQ06 | 2 | |  | |  | |  | |  | |  | |  | | X | |  | | X | |
| REQ07 | 3 | |  | |  | |  | |  | |  | |  | |  | | X | |  | |
| REQ08 | 4 | |  | | X | | X | |  | |  | |  | |  | |  | |  | |
| REQ09 | 1 | |  | |  | |  | |  | |  | | X | |  | |  | |  | |
| Max PW | | 3 | | 4 | | 4 | | 2 | | 2 | | 1 | | 2 | | 3 | | 3 | |
| Total PW | | 5 | | 9 | | 7 | | 2 | | 2 | | 1 | | 2 | | 5 | | 8 | |

טבלה 12 traceability matrix use case vs requirements

3.8 **Sequence Diagram**

תמונה שמכילה שולחן

התיאור נוצר באופן אוטומטי

איור 4 SEQ- תהליך המרה

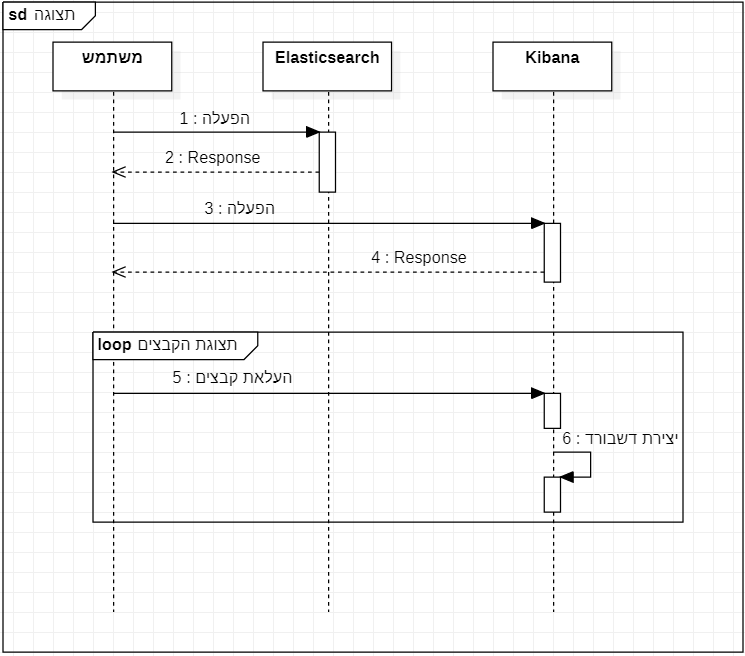
תהליך זה מתאר את תהליך ההמרה:

* בשלב הראשון, המערכת תבצע הפעלה למערכת.
* בשלב השני, המערכת תבצע קריאה לקובץ ההגדרות.
* בשלב השלישי, המערכת תקבל את קובץ ההגדרות.
* בשלב הרביעי, המערכת תבצע התחברות למסד הנתונים
* בשלב החמישי, המערכת תקבל את כל הדוחות של המכונות ממסד הנתונים.
* בשלב השישי, המערכת תבצע הפעלה של תהליך ההמרה המתבצע כמספר המכונות שהתקבלו בשלב החמישי.
* בכל לולאה המערכת תבצע מספר שלבים:

- בדיקת תקינות הקובץ (שלב שביעי)

- ביצוע טרנספורמציה (שלב שמיני)

- שמירת הדוח בפורמט הנבחר (שלב תשיעי)

איור 5 SEQ- תהליך ההצגה

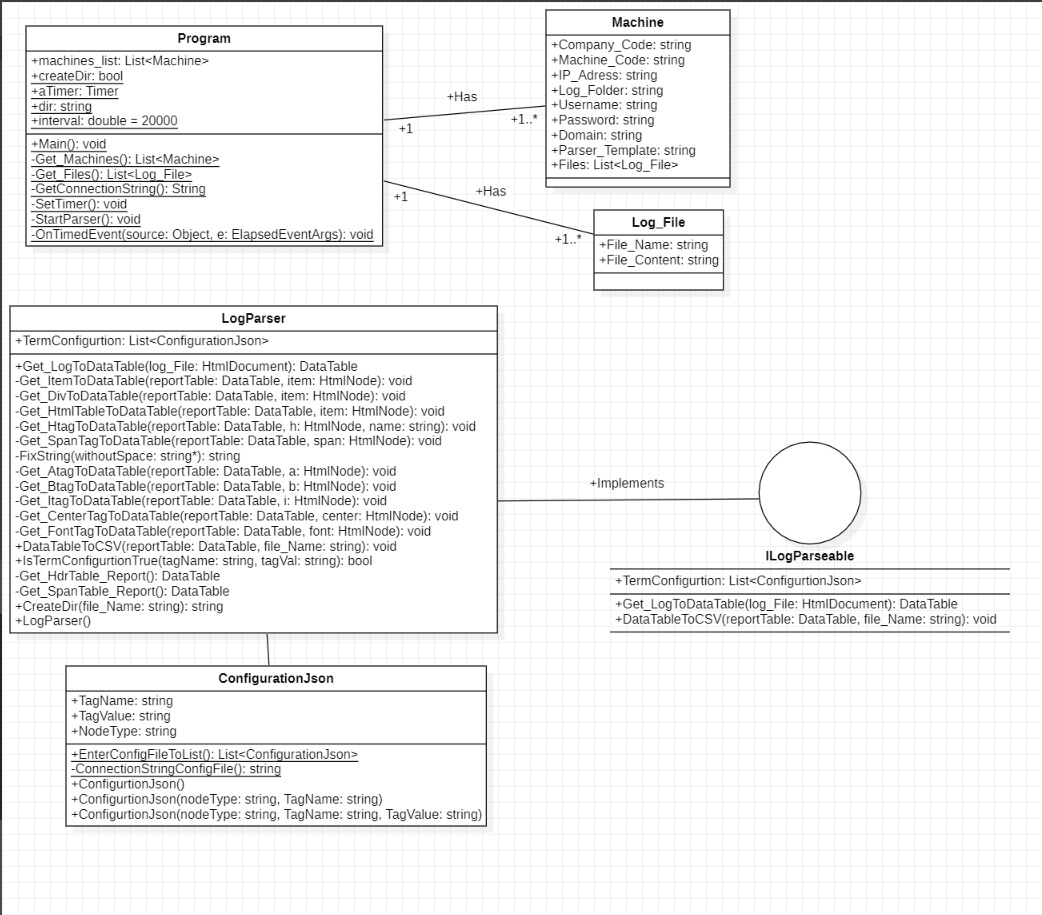
תהליך זה מתאר את תהליך ההצגה:

* בשלב הראשון, המערכת תבצע הפעלה לElasticsearch.
* בשלב השני, המערכת תקבל תשובה האם ההפעלה בוצעה בהצלחה.
* בשלב השלישי, המערכת תבצע הפעלה ל.Kibana
* בשלב הרביעי, המערכת תקבל תשובה האם ההתחברות בוצעה בהצלחה
* בשלב החמישי, המערכת תבצע הפעלה של תהליך תצוגת הקבצים כמספר הקבצי CSV שהמשתמש רוצה
* בכל לולאה המערכת תבצע מספר שלבים:

- העלאת קבצים(שלב חמישי)

- יצירת דשבורד (שלב שישי)

3.9 **Class Diagram**



איור 6 – Class Diagram

3.10 **המחשת המערכת**

קובץ הקונפיגורציה מורכב ממערך אובייקטים של JSON:

* nodeType – המקבל ערכים: HTML/file. נועד להבדיל בין תג HTML לבין מיקום הקבצים. במידה ומדובר בתג html זה אומר שעלינו לבצע תרגום, במידה ומדובר בקובץ file תפתח תיקיה עם הלוגים.
* tagName –במידה ומדובר בערך של HTML בשדה זה נרשם שם התג של הHTML. במידה ומדובר בfile יהיה בו ערך ברירת מחדל: fileLocation
* tagValue – במידה ומדובר בערך של Html, בשדה נרשם האם נרצה לבצע המרה לפי תנאי כלשהו. במידה ומדובר בערך של ,file בשדה זה נרשם המיקום בו ישמר הקובץ לאחר המרה.

דוגמא לאובייקט המקבל ערך של HTML, תג של table ואין תנאי נוסף:

תמונה שמכילה טקסט

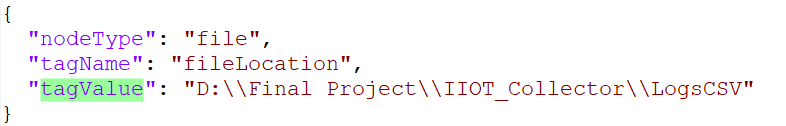
התיאור נוצר באופן אוטומטי

דוגמא לאובייקט המקבל ערך של HTML, תג של h5 ויש בו תנאי האם המילה test קיימת בתג זה:

תמונה שמכילה טקסט

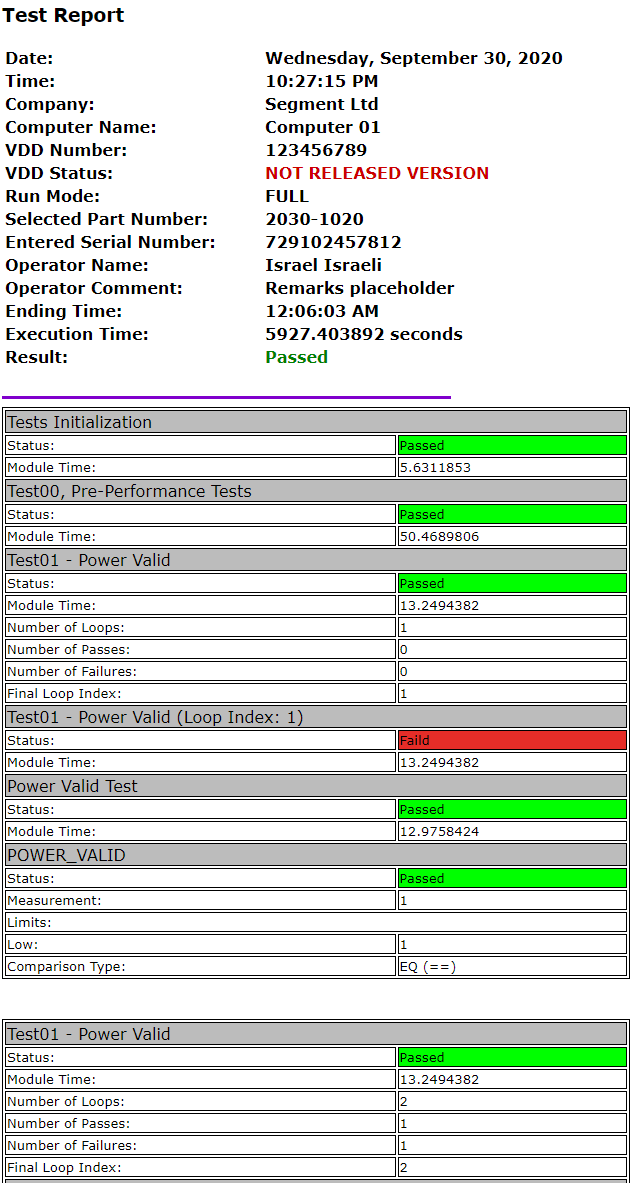
התיאור נוצר באופן אוטומטי

דוגמא לאובייקט המקבל ערך של file , ערך ברירת המחדל ומיקומו:

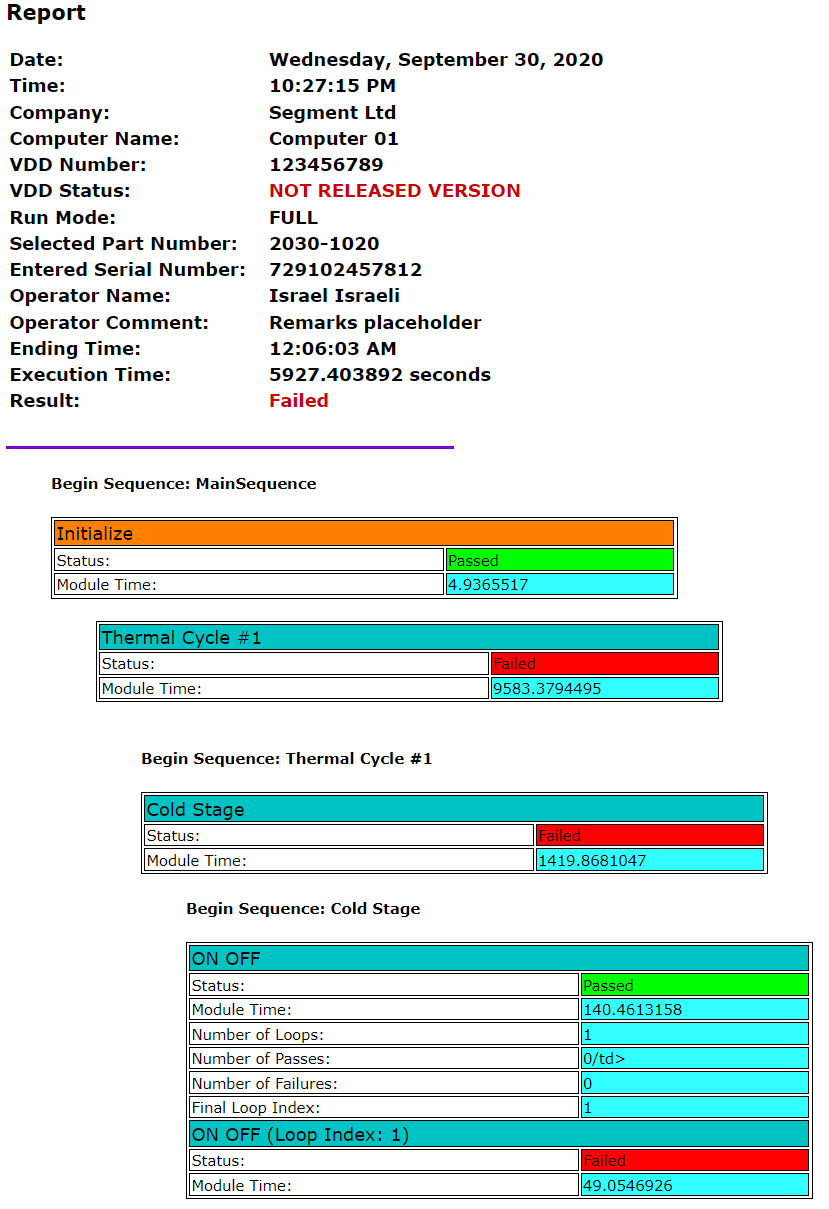


כאשר המערכת מבצעת התחברות למסד הנתונים היא מקבלת את כל הדוחות של המכונות. אצלנו במערכת יש 3 מכונות ודוחות המכונות מתקבלות בצורה הזו:

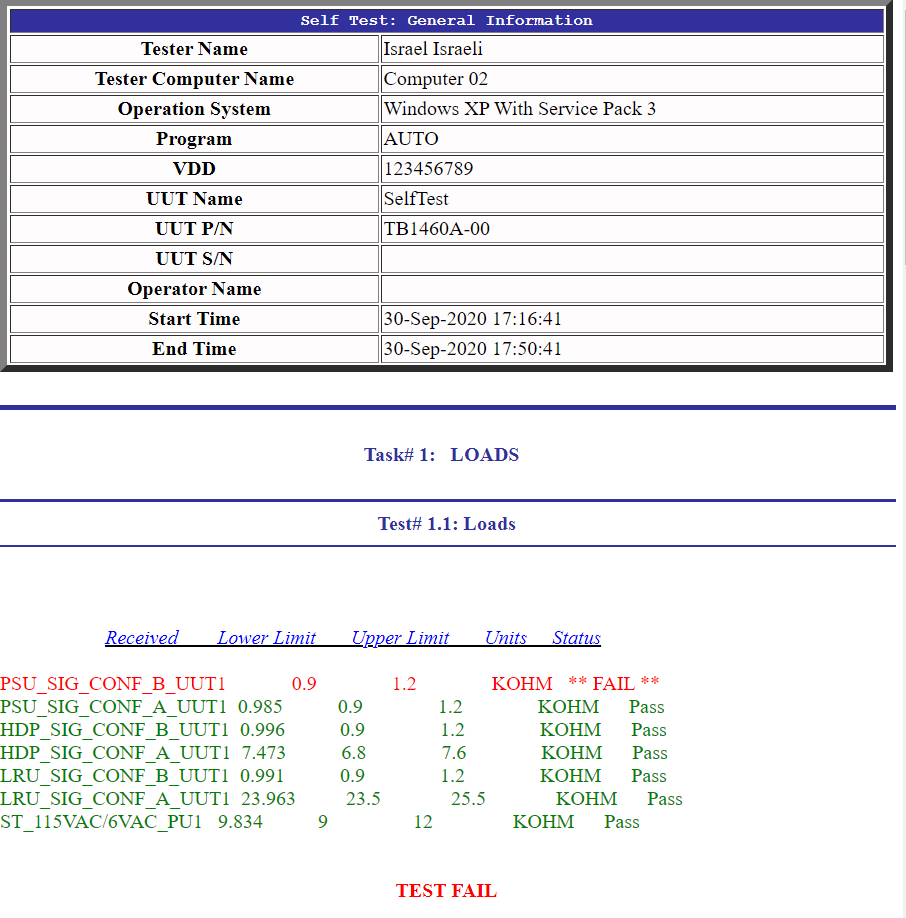
דוח עבור מכונה מס' 1:



דוח עבור מכונה מס' 2:

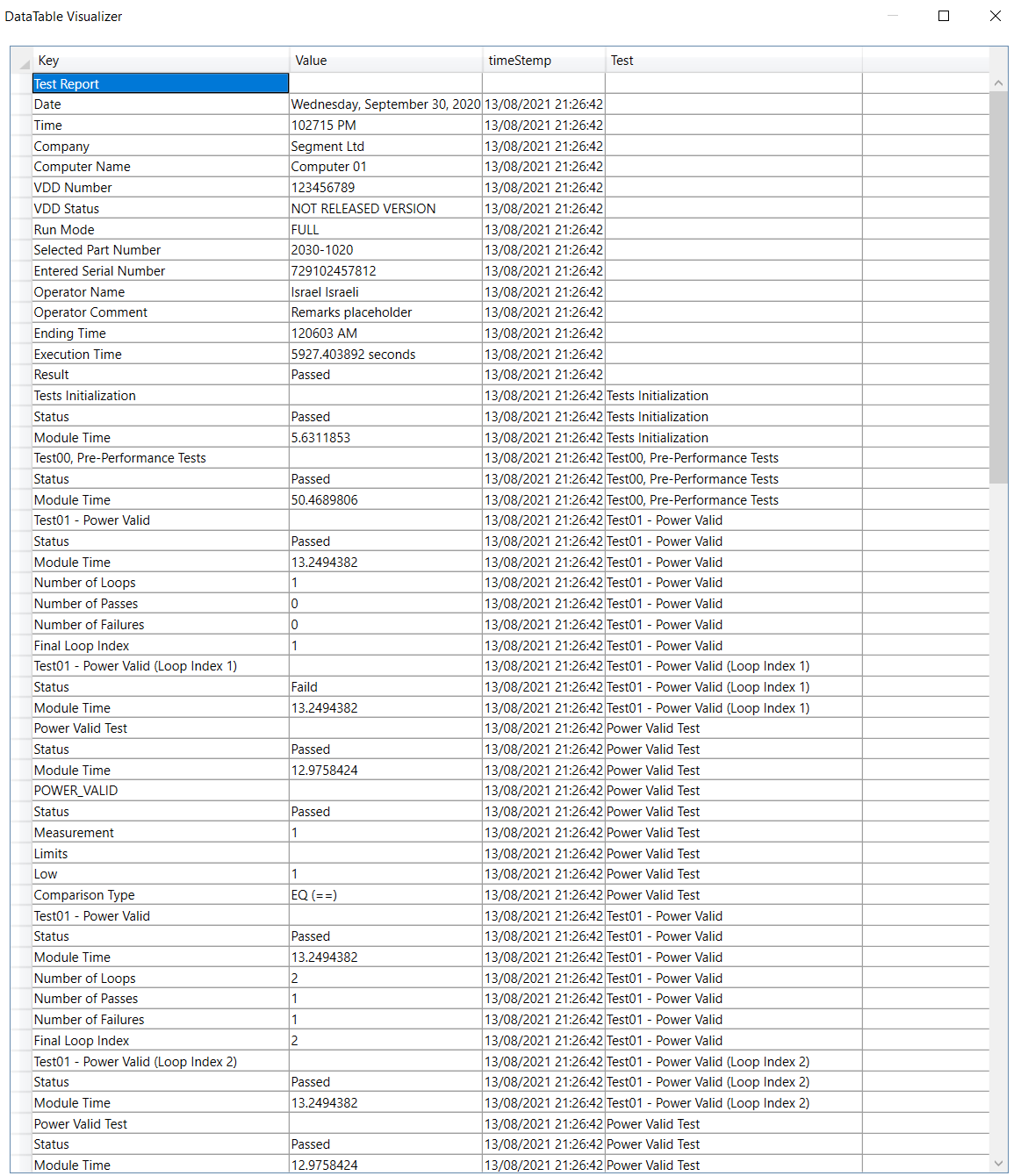


דוח עבור מכונה מס' 3:

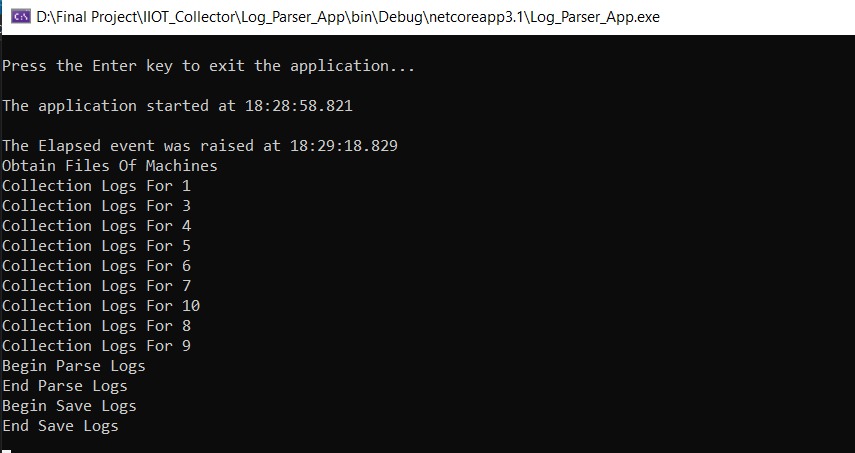


כל דוח מתקבל בצורה שונה.

בתהליך ההמרה, המערכת שומרת כל דוח בטבלה מקומית:



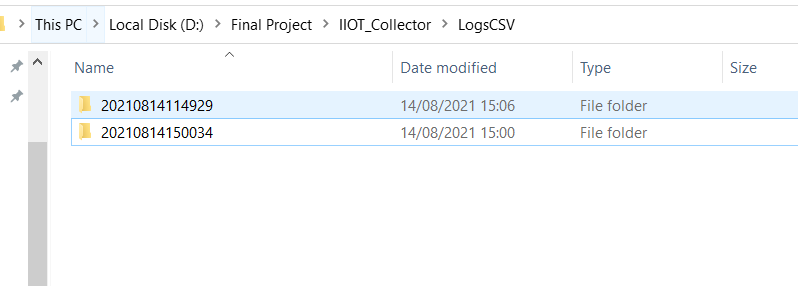
לאורך כל תהליך ההמרה יש אינדיקציה למשתמש באיזה שלב הוא נמצא:



4. **תוצאות הבדיקה והדגמות**

עבור כל מכונה התקבלו מס' לוגים. עבור כל לוג התקבל קובץ CSV אותו ניתן להכניס לELK .

עבור כל ריצה נפתחת תיקיה חדשה. שם התיקיה נקבע לפי תאריך ושעון UTC0.



בכל תיקיה ניתן לראות שעבור כל מכונה התקבלו מספר לוגים. כל קובץ לוג הומר לCSV. אצלנו במערכת יש שני סוגי של תצורת הלוגים: )Sample\_Log\_01 היכול להקרא גם Sample\_Log) ן Sample\_Log\_02.

מספר המכונה מסומן בצהוב.

עבור כל מכונה התקבלו מספר הלוגים. מסומן בכחול לדוגמא.

תמונה שמכילה שולחן

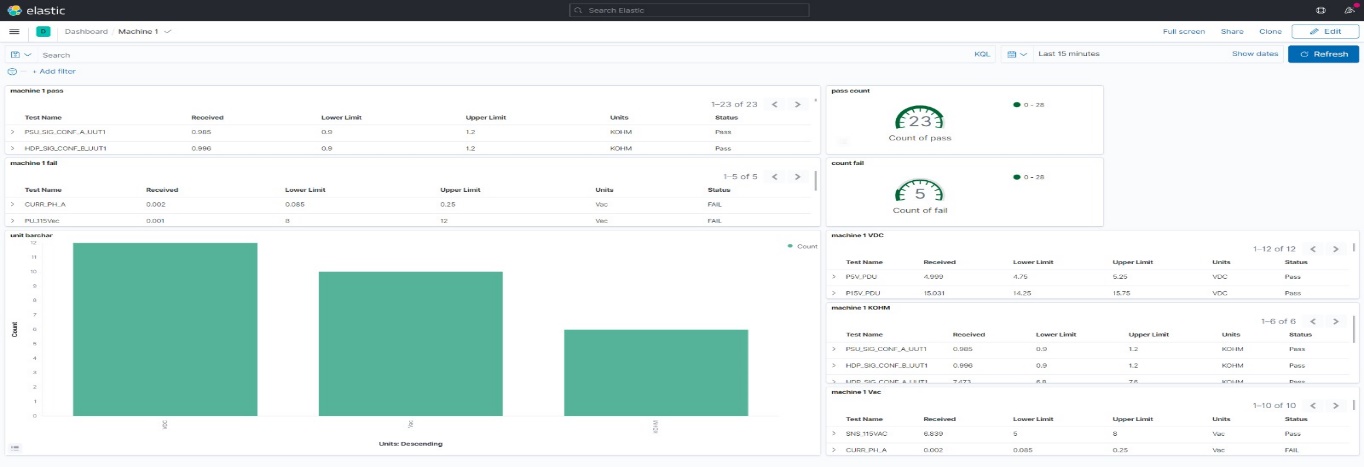
התיאור נוצר באופן אוטומטי



עבור שני סוגי הלוגים התקבלו שתי תצוגות שונות בELK:

מכונה 1:

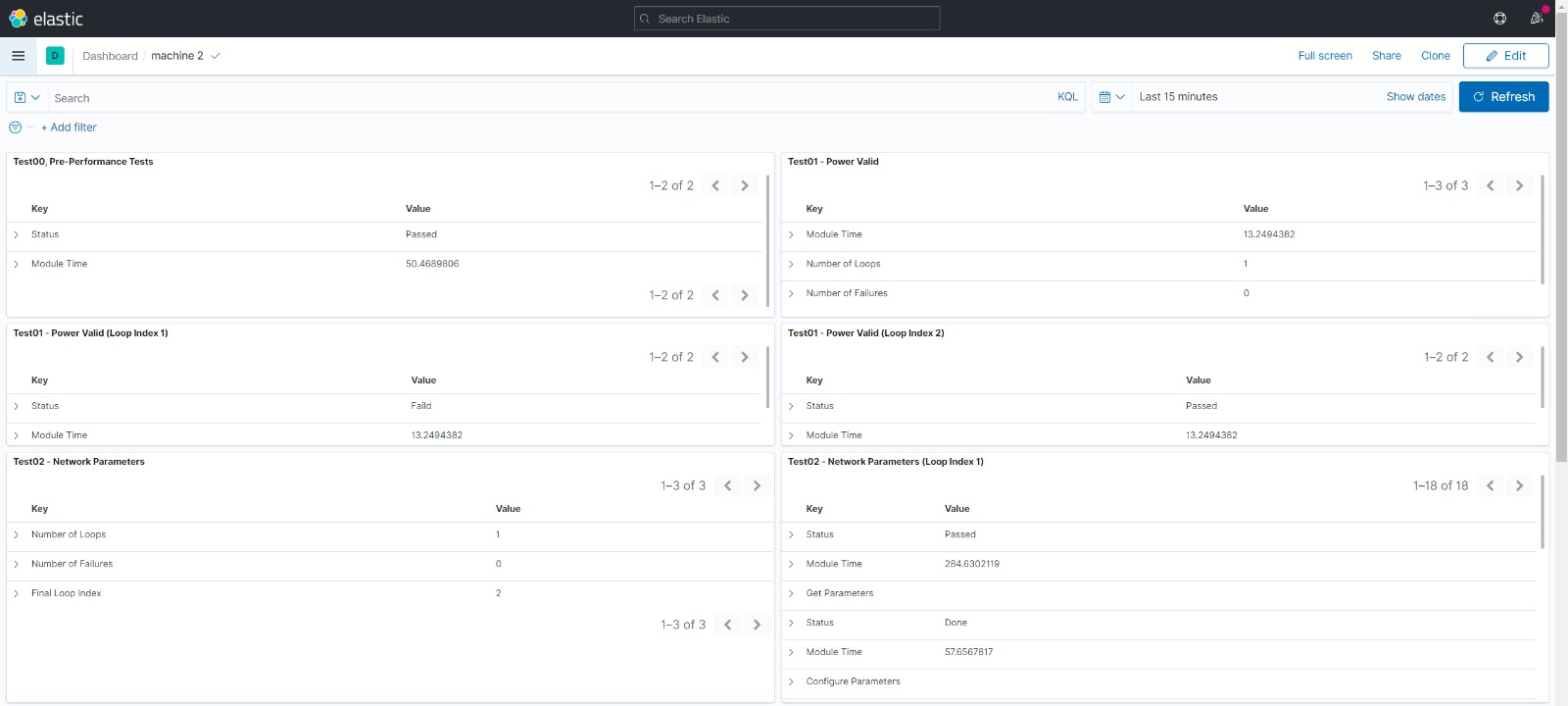
ניתן לראות פרטים על: מבחני הצלחה וכשלון, כמה עברו בהצלחה וכמה נכשלו ומידע על מבדקים שונים.



איור 7 machine 1 Kibana panel

מכונה 2:

ניתן לראות פרטים על כל מבדק שבוצע במכונה.



איור 8 machine 2 Kibana panel

5. **מסקנות והמלצות**

פרויקט זה התמקד בפיתוח מערכת העובדת כתהליך רקע ומספקת למשתמש ניתוח מפורט ומהימן אודות פעילות המכונות באופן שוטף וללא תלות בסוגי הלוגים השונים שמיוצרים על ידי המכונות.

אפיון הדרישות שנעשה עבור המערכת כלל בתוכו דרישות שונות:

* דרישה 1: לפני כל הטמעה של המערכת במפעל תתבצע בדיקה שאכן המכונות מעלות באופן עצמאי את הדוחות.
* דרישה 2: סופק קובץ הגדרות בפורמט json המספק יכולות דינאמיות לתרגום הדוחות במערכת.
* דרישה 3: סופקה קריאה של קובץ ההגדרות על ידי המערכת המתוזמנת לפעם ביום.
* דרישה 4: המידע במערכת קריא ונגיש.
* דרישה 5: המידע מהלוגים מתורגם בעת הפעלה המערכת ומתוזמן לפעם ביום.
* דרישה 6: המערכת מציגה את הנתונים על גבי ממשק נוח וקריא. על המשתמש להעלות את הקבצים לELK ולייצר ממשק תצוגה על פי בחירתו.
* דרישה 7: למשתמש יש אפשרות לשנות את קובץ ההגדרות לפי בחירתו.
* דרישה 8: לוגים אינם תקינים לא מתורגמים על ידי המערכת.
* דרישה 9 : המידע המתקבל מהלוגים מספק מידע אמין אודות מצב המוצרים שהמכונות בדקו.

מתוצאות אלו ניתן לראות שרוב הדרישות נענו. בנוסף יש צורך בהוספת מנגנון אשר יכניס באופן אוטומטי את קבצי הcsv בהם קיים המידע המתורגם לELK.

במהלך הפרויקט חזינו סיכונים:

1. חשש שהלוגים המתקבלים אינם תקינים ואפשריים לתרגום- הגדרנו את סיכון זה ברמת סיכון 5 (הגבוהה ביותר), משום שסיכון זה מהווה חלק מהותי במערכת. סיכון זה נמנע על ידי כך שבזמן ריצת המערכת במידה ונתקל בלוג לא תקין, המערכת תדע להפסיק את התרגום ובכך להימנע מכשל המערכת.

2. קושי בהבנה של מהו לוג תקין ומה לא ומה ההבדל בניהם – הגדרנו את סיכון זה ברמת סיכון 5 (הגבוהה ביותר), רמת סיכון זאת נבחרה משום שלא היה לנו ידע קודם בהגדרה הנכונה לפעילות תקינה של מכונות מה שעלול להשפיע על היכולת של המערכת לבצע תרגום נכון. סיכון זה נמנע ע"י חקירה נכונה של הלוגים.

3. אי עמידה בזמנים עקב עומס בלימודים- הגדרנו את סיכון זה ברמת סיכון 4. הסיכון התממש חלקית תוך שילוב עם סיכון נוסף אותו לא חזינו.

הסיכון אותו לא חזינו אך במהלך פיתוח הפרויקט התברר לנו כבעל השפעה היה ביטול ההתקשרות עם החברה החיצונית. בתחילת סמסטר א' הפרויקט נעשה בשיתוף עם חברה חיצונית ולאחר כחצי שנה(סוף סמסטר א') הקשר עם החברה בוטל. דבר זה הוביל לשינוי פיתוח הפרויקט, אופי הפרויקט, תוספת עבודה שדרשה מאיתנו תכנון זמנים שונים והספק מהיר יותר על מנת לסיים את הפרויקט בזמן. שינוי זה הוביל לכך שהדרישה השלישית לא סופקה במלואה.

בנוסף הפרויקט בוצע בתקופת הקורונה. המפגשים הקבוצתיים במהלך הפרויקט שלנו בוצעו באמצעות פגישות זום. בדיעבד היינו משנים את אופן העבודה בכך שהיינו מבצעים מפגשים פיזיים דבר היה תורם לחיבור נוסף, לסביבת תומכת למידה ומחדד את האמון בנינו.

לסיכום, יצירת לוח זמנים משותף הכולל את האילוצים של כל אחד מחברי הצוות ויצירת רשימות של משימות ועדיפותן תרם לנו להתקדמות יום יומית בפרויקט, השגת היעדים בשלמותם והתגברות על הסיכון שהוזכר למעלה.

6. **רשימת מקורות**

<https://www.learnentityframeworkcore.com/>

<https://www.mssqltips.com/sqlservertip/5771/querying-sql-server-tables-from-net/>

<https://docs.microsoft.com/en-us/dotnet/api/system.data.sqlclient.sqlconnection?view=netframework-4.7.2>

<https://docs.microsoft.com/en-us/aspnet/core/fundamentals/http-requests?view=aspnetcore-5.0>

<https://www.newtonsoft.com/json/help/html/SerializeObject.htm>

<https://html-agility-pack.net/documentation>

<https://www.guru99.com/c-sharp-access-database.html>

<https://github.com/Dynalon/JsonConfig>

<https://www.w3schools.com/js/js_json_intro.asp>

<https://www.json.org/json-en.html>

<https://www.youtube.com/channel/UC7z5VlhDHnorjUm6oW5dXcw/>

<https://www.elastic.co/guide/index.html>

<https://www.elastic.co/start>

7. **Abstract**

The purpose of the system is to transform the information obtained from production machines and testing equipment that produce logs into knowledge, i.e., to make the information readable and accessible and thus to form a basis for information management and mining. Information mining is the collection of reports on machines that contain information about their output.

The system allows the definition of processes for collecting the information from the machines, performing processing operations on the information for the purpose of obtaining a uniform configuration and saving it for further research.

In fact, our system integrates all the reports received from the machines and transforms them to suit a format capable of interfacing with an external server.

The manner in which the logs are received, arranged and maintained is carried out at a regular time, once a day, thus providing reliable information about the condition of the machines and their operation.

The main role of the system is to extract all the logs obtained from the various testing machines in different formats, perform a transformation on them and present all these reports to the users.

The construction of this system includes 2 subsystems:

HTML page analysis system that performs retrieval, compatibility, transition and scheduling.

A display system that performs analysis and display.

First the system receives a configuration file (JSON file) specifying for which HTML tags the information should be translated. The system first performs retrieval of logs from the database to which the machines are connected, the logs obtained from the database are in HTML format and each report is different in form from the other. The system then transforms them using an HTML Agility library whose job is to identify HTML tags and for each tag displays all of its attributes in a designated class. The system will insert data into the DataTable (local storage in the system).

Once this operation has been done, the system will save the information in CSV (Comma-Separated Values) format in a local folder according to the configuration file. The system will then transmit the information received to the display system. This system will receive the CSV file and will allow the reports to be displayed to users in an accessible and orderly manner on top of a system called ELK (ELastic search and Kibana).

This is a free system that includes a collection of open-source projects that allow analysis and display of logs on visual panels. It allows analysis and search of the existing information in various formats: JSON (java script object notation) and CSV (Comma-Separated Values).

This project focused on developing a system that works as a background process and provides the user with a detailed and reliable analysis of the operation of the machines on an ongoing basis and regardless of the type of machine / machines in the various factories or the different logs produced by them, in order to provide a solution for factories with old testing machines that produce logs in different and uneven formats. The system provides a correct process of retrieval and compatibility, information conversion, good transmission and analysis and display in a readable and understandable way to the user.

8. **נספחים**

טבלת סיכונים המתארת את הסיכונים השונים העלולים להתממש במהלך תכנון ובניית הפרויקט.

|  |  |  |  |
| --- | --- | --- | --- |
| תיאור הסיכון | רמת הסיכון | סבירות | פעילות מונעת לצמצום הסיכון |
| הלוגים המתקבלים לא יהיו תקינים | 5 | 2 | לתת למכונה לסיים את הפעולה בצורה תקינה לפני שנפעיל את המערכת שלנו |
| קושי בהבנה בהבדל בין לוג תקין לבין לוג שאינו תקין | 5 | 2 | לפני ביצוע המערכת שלני תעשה בדיקה ברמת המכונה |
| אי עמידה בזמנים עקב עומס בלימודים | 4 | 4 | חלוקת עבודה וניהול זמנים תוך התחשבות בקורסים נוספים הנלמדים במקביל לפרויקט |

טבלה 13 ניהול סיכונים

קובץ ההגדרות(קונפיגורציה)- JsonConfig.json :

[

{

"nodeType": "html",

"tagName": "table",

"tagValue": null

},

{

"nodeType": "html",

"tagName": "h3",

"tagValue": "Test"

},

{

"nodeType": "html",

"tagName": "h5",

"tagValue": "Test"

},

{

"nodeType": "html",

"tagName": "a",

"tagValue": null

},

{

"nodeType": "html",

"tagName": "b",

"tagValue": null

},

{

"nodeType": "html",

"tagName": "i",

"tagValue": null

},

{

"nodeType": "html",

"tagName": "center",

"tagValue": null

},

{

"nodeType": "html",

"tagName": "font",

"tagValue": null

},

{

"nodeType": "html",

"tagName": "span",

"tagValue": null

},

{

"nodeType": "file",

"tagName": "fileLocation",

"tagValue": "D:\\Final Project\\IIOT\_Collector\\LogsCSV"

}

]

**קוד:**

מחלקת **Program** :

מכילה את הmain של המערכת.

using HtmlAgilityPack;

using System;

using System.Collections.Generic;

using System.Data;

using System.Data.SqlClient;

using System.IO;

using System.Threading.Tasks;

using Log\_Parser\_App.Log\_Sample;

using System.Timers;

namespace Log\_Parser\_App

{

class Program

{

private static Timer aTimer;

public static List<Machine> machines\_list = new List<Machine>();

public static bool createDir;

public static string dir;

private static double interval = 20000;

static void Main(string[] args)

{

SetTimer();

Console.WriteLine("\nPress the Enter key to exit the application...\n");

Console.WriteLine("The application started at {0:HH:mm:ss.fff}\n", DateTime.Now);

Console.ReadLine();

aTimer.Stop();

aTimer.Dispose();

Console.WriteLine("Terminating the application...");

}

/\*SetTimer to how much time delay the program\*/

private static void SetTimer()

{

// Create a timer with a two second interval.

aTimer = new System.Timers.Timer(interval);

// Hook up the Elapsed event for the timer.

aTimer.Elapsed += OnTimedEvent;

aTimer.AutoReset = true;

aTimer.Enabled = true;

}

/\*OnTimedEvent the event that happend in the end of the timer\*/

private static void OnTimedEvent(Object source, ElapsedEventArgs e)

{

Console.WriteLine("The Elapsed event was raised at {0:HH:mm:ss.fff}",

e.SignalTime);

StartParser();

}

/\*StartParser- start the parsering process\*/

private static void StartParser()

{

createDir = true;

DataTable reportTable;

HtmlDocument log = new HtmlDocument();

LogParser parser = new LogParser();

if (parser.TermConfigurtion == null)

Console.WriteLine("No Configurtion file");

else

{

machines\_list = Get\_Machines();

Console.WriteLine("Obtain Files Of Machines");

Parallel.ForEach(machines\_list, (current\_machine) =>

{

Console.WriteLine("Collection Logs For " + current\_machine.Machine\_Code);

current\_machine.Files = Get\_Files(current\_machine);

});

Console.WriteLine("Begin Parse Logs");

foreach (Machine machine in machines\_list)

{

foreach (Log\_File log\_File in machine.Files)

{

log.LoadHtml(log\_File.File\_Content);

reportTable = parser.Get\_LogToDataTable(log);

parser.DataTableToCSV(reportTable, log\_File.File\_Name.Replace(".html", "") + "\_" + machine.Machine\_Code + "\_" + DateTime.Now.ToString("yyyyMMddHHmm"));

}

}

Console.WriteLine("End Parse Logs");

Console.WriteLine("Begin Save Logs");

Console.WriteLine("End Save Logs");

}

}

/\*Get\_Machines-Bring the machines from the DB\*/

private static List<Machine> Get\_Machines()

{

List<Machine> Local\_List = new List<Machine>();

try

{

string connectionString = GetConnectionString();

using (SqlConnection mycon = new SqlConnection(connectionString))

{

mycon.Open();

using (SqlCommand mycmd = new SqlCommand())

{

mycmd.CommandTimeout = 0;

mycmd.Connection = mycon;

mycmd.CommandType = CommandType.Text;

mycmd.CommandText = "SELECT Company\_Code, Machine\_Code, IP\_Adress, Log\_Folder, Username, Password, Domain, Key\_Parsers.Parser\_Template FROM Key\_Machines inner join Key\_Parsers on Key\_Machines.Parser\_Code = Key\_Parsers.Parser\_Code WHERE Machine\_Type=2 ORDER BY Company\_Code ASC";

SqlDataReader reader = mycmd.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

Machine machine = new Machine();

machine.Company\_Code = reader["Company\_Code"].ToString();

machine.Machine\_Code = reader["Machine\_Code"].ToString();

machine.IP\_Adress = reader["IP\_Adress"].ToString();

machine.Log\_Folder = reader["Log\_Folder"].ToString();

machine.Username = reader["Username"].ToString();

machine.Password = reader["Password"].ToString();

machine.Domain = reader["Domain"].ToString();

machine.Parser\_Template = reader["Parser\_Template"].ToString();

Local\_List.Add(machine);

}

}

}

mycon.Close();

}

}

catch (Exception ex)

{

Console.WriteLine("Error: " + ex.InnerException);

}

return Local\_List;

}

/\*Get\_Files-Brings the LOGS from the folder they are stored\*/

private static List<Log\_File> Get\_Files(Machine machine)

{

List<Log\_File> Files = new List<Log\_File>();

var result = System.IO.Directory.GetFiles(@"\\" + machine.IP\_Adress + machine.Log\_Folder);

foreach (string Log\_File in result)

{

Log\_File Current\_File = new Log\_File();

Current\_File.File\_Name = Path.GetFileName(Log\_File);

Current\_File.File\_Content = File.ReadAllText(Log\_File);

Files.Add(Current\_File);

}

return Files;

}

/\*GetConnectionString-Connection String to the DB for security reseaon\*/

static private string GetConnectionString()

{

// To avoid storing the connection string in your code,

// you can retrieve it from a configuration file.

return "Data Source =DESKTOP-N5JUU5Q\\SHAKEDDB; Initial Catalog = IIOT\_Collector; Integrated Security = True; Connect Timeout = 30; Encrypt = False; TrustServerCertificate = False; ApplicationIntent = ReadWrite; MultiSubnetFailover = False";

}

}

}

מחלקת **ILogParser**:

using System;

using System.Collections.Generic;

using System.Text;

using System.Data;

using HtmlAgilityPack;

namespace Log\_Parser\_App

{

/\*Interface for abstarcting the the program structure and

Simplifies if in the future we are asked to expand the system \*/

interface ILogParser

{

/\*Get the log to a DataTable class\*/

public DataTable Get\_LogToDataTable(HtmlDocument log\_File);

/\*Parser the the DataTable to CSV\*/

public void DataTableToCSV(DataTable reportTable, string file\_Name);

}

}

מחלקת **ILogParser**:

using HtmlAgilityPack;

using System;

using System.Collections.Generic;

using System.Data;

using System.Linq;

using System.Text;

using System.IO;

using System.Text.RegularExpressions;

using Nest;

namespace Log\_Parser\_App.Log\_Sample

{

/\*LogParser implement the interface and parsering the HTML file to CSV \*/

class LogParser :ILogParser

{

/\*List<ConfigurtionJson> for holding the Configurtion from the Configurtion file,

while the system running\*/

public List<ConfigurtionJson> TermConfigurtion;

/\*NO Arguments Constructor\*/

public LogParser()

{

this.TermConfigurtion = ConfigurtionJson.EnterConfigFileToList();

}

/\*Parsering the HTML file to HtmlNodes and sorting with function Get\_ItemToDataTable\*/

public DataTable Get\_LogToDataTable(HtmlDocument log\_File)

{

if (log\_File != null)

{

if (this.TermConfigurtion.Any())

{

string ClassToGet = "hdr";

DataTable reportTable;

List<HtmlNode> divs = log\_File.DocumentNode.SelectSingleNode("//body").DescendantsAndSelf().Where(x => x.NodeType == HtmlNodeType.Element && x.ParentNode.Name == "body").ToList();

divs.RemoveAll(x => x.Name == "br" || x.Name == "hr");

if (divs[1].SelectNodes("//table[@class='" + ClassToGet + "']") != null)

reportTable = Get\_HdrTable\_Report();

else

reportTable = Get\_SpanTable\_Report();

foreach (HtmlNode item in divs)

{

Get\_ItemToDataTable(reportTable, item);

}

return reportTable;

}

else

{

Console.WriteLine("No Configurtion file");

return null;

}

}

return null;

}

/\*Checking the HTMLNode and determines which type is the node and calling the right

\* function and checking the term from the Configuration file\*/

private void Get\_ItemToDataTable(DataTable reportTable, HtmlNode item)

{

string testName="1";

if (item.Name == "table" && IsTermConfigurtionTrue("table"))

Get\_HtmlTableToDataTable(reportTable, item);

if (item.Name == "h3" && IsTermConfigurtionTrue("h3") || item.Name == "h5" && IsTermConfigurtionTrue("h5"))

Get\_HtagToDataTable(reportTable, item, testName);

if (item.Name == "div")

Get\_DivToDataTable(reportTable, item);

if (item.Name == "span" && IsTermConfigurtionTrue("span"))

Get\_SpanTagToDataTable(reportTable, item);

if (item.Name == "a" && IsTermConfigurtionTrue("a"))

Get\_AtagToDataTable(reportTable, item);

if (item.Name == "b" && IsTermConfigurtionTrue("b"))

Get\_BtagToDataTable(reportTable, item);

if (item.Name == "center" && IsTermConfigurtionTrue("center"))

Get\_CenterTagToDataTable(reportTable, item);

if (item.Name == "i" && IsTermConfigurtionTrue("i"))

Get\_ItagToDataTable(reportTable, item);

}

/\*Extracting the informtion from the Div tag to the DataTable\*/

private void Get\_DivToDataTable(DataTable reportTable, HtmlNode item)

{

List<HtmlNode> divChilds = item.ChildNodes.Where(x => x.NodeType == HtmlNodeType.Element).ToList();

foreach (HtmlNode divChild in divChilds)

{

Get\_ItemToDataTable(reportTable, divChild);

}

}

/\*Extracting the informtion from the Table tag to the DataTable\*/

private void Get\_HtmlTableToDataTable(DataTable reportTable, HtmlNode item)

{

string timeStemp = DateTime.Now.ToString();

string testName="";

string[] check2;

string check3 = "";

List<HtmlNode> tables = item.ChildNodes.Descendants().Where(x => x.NodeType == HtmlNodeType.Element && x.ParentNode.Name == "tbody").ToList();

HtmlNode lastTd;

HtmlNode elementInTd;

foreach (HtmlNode tr in tables)

{

List<HtmlNode> tds = tr.Descendants().Where(x => x.NodeType == HtmlNodeType.Element && x.ParentNode.Name == tr.Name).ToList();

lastTd = tds.LastOrDefault();

foreach (HtmlNode td in tds)

{

if (td.ParentNode.Name != "font")

{

elementInTd = td.DescendantsAndSelf().Where(x => x.NodeType == HtmlNodeType.Element && x.ParentNode.Name == td.Name).SingleOrDefault();

if (elementInTd != null)

{

if (elementInTd.ChildNodes.Any(x => x.NodeType == HtmlNodeType.Element && x.ParentNode.Name == elementInTd.Name))

elementInTd = elementInTd.ChildNodes.Where(x => x.NodeType == HtmlNodeType.Element).SingleOrDefault();

}

else

{

elementInTd = td;

}

check3 += elementInTd.InnerHtml.Trim().Replace(":", "").Replace("&nbsp;", "");

check3 = Regex.Replace(check3, @"(\<(\/)?(\w)\*(\d)?\>)", string.Empty);

if (td != lastTd)

check3 += "&&&";

if (elementInTd.Attributes.Count()>0)

if (elementInTd.InnerHtml.Contains("Test"))

testName = check3.Split("&&&")[0];

else if(reportTable.AsEnumerable().Last().ItemArray[3].ToString() != "")

testName = reportTable.AsEnumerable().Last().ItemArray[3].ToString();

}

else

Get\_FontTagToDataTable(reportTable, td);

}

if(check3.Split("&&&").Length== 2)

check3 = check3 + "&&&" + timeStemp+ "&&&" + testName;

else

check3 = check3 + "&&&" +" "+ "&&&" + timeStemp+ "&&&" + testName;

check2 = check3.TrimEnd().Split("&&&");

reportTable.Rows.Add(check2);

check3 = "";

}

}

/\*Extracting the informtion from the H tag to the DataTable\*/

private void Get\_HtagToDataTable(DataTable reportTable, HtmlNode h,string testName)

{

ConfigurtionJson checkTerm;

if (h.Name == "h3")

{

checkTerm = this.TermConfigurtion.Find(tc => tc.TagName == "h3");

if(checkTerm.TagValue!=null && h.InnerText.Trim().Contains(checkTerm.TagValue))

reportTable.Rows.Add(h.InnerText.Trim());

else if(checkTerm.TagValue == null)

reportTable.Rows.Add(h.InnerText.Trim());

testName = null;

}

if (h.Name == "h5")

{

checkTerm = this.TermConfigurtion.Find(tc => tc.TagName == "h5");

if (checkTerm.TagValue != null && h.InnerText.Trim().Contains(checkTerm.TagValue))

reportTable.Rows.Add(h.InnerText.Trim(),"",DateTime.Now, h.InnerText.Trim());

else if (checkTerm.TagValue == null)

reportTable.Rows.Add(h.InnerText.Trim(), "", DateTime.Now, h.InnerText.Trim());

testName = h.InnerText.Trim();

}

}

/\*Extracting the informtion from the Span tag to the DataTable\*/

private void Get\_SpanTagToDataTable(DataTable reportTable, HtmlNode span)

{

if (span.Descendants().Where(x => x.NodeType == HtmlNodeType.Element && x.ParentNode.Name == span.Name).Count() == 0)

{

string check = span.InnerText.Replace("&nbsp;", " ").Replace("\*\*","").Trim();

check = Regex.Replace(check, @"\s+", " ").Trim();

string[] withoutSpace = check.Split(" ");

if (withoutSpace.Length >6)

{

withoutSpace[0] = FixString(withoutSpace);

reportTable.Rows.Add(withoutSpace[0]);

check = " ";

}

if (check.Length > 0 && check!=" ")

reportTable.Rows.Add(withoutSpace);

}

else

{

List<HtmlNode> spanChildren = span.ChildNodes.Where(x => x.NodeType == HtmlNodeType.Element).ToList();

foreach (HtmlNode spanChild in spanChildren)

{

if (spanChild.Name != "br")

Get\_ItemToDataTable(reportTable, spanChild);

}

}

}

/\*Extracting the informtion from the a tag to the DataTable\*/

private void Get\_AtagToDataTable(DataTable reportTable, HtmlNode a)

{

List<HtmlNode> aChildren = a.ChildNodes.Where(x => x.NodeType == HtmlNodeType.Element).ToList();

foreach (HtmlNode item in aChildren)

{

Get\_ItemToDataTable(reportTable, item);

}

}

/\*Extracting the informtion from the b tag to the DataTable\*/

private void Get\_BtagToDataTable(DataTable reportTable, HtmlNode b)

{

reportTable.Rows.Add(b.InnerText.Replace("&nbsp;", " ").Trim());

}

/\*Extracting the informtion from the i tag to the DataTable\*/

private void Get\_ItagToDataTable(DataTable reportTable, HtmlNode i)

{

List<HtmlNode> iChildren = i.Descendants().Where(x => x.NodeType == HtmlNodeType.Element && x.ParentNode.Name == i.Name).ToList();

foreach (HtmlNode iChild in iChildren)

{

Get\_ItemToDataTable(reportTable, iChild);

}

}

/\*Extracting the informtion from the Center tag to the DataTable\*/

private void Get\_CenterTagToDataTable(DataTable reportTable, HtmlNode center)

{

List<HtmlNode> centerChildren = center.Descendants().Where(x => x.NodeType == HtmlNodeType.Element).ToList();

string check = "";

int count = 0;

foreach (HtmlNode centerChild in centerChildren)

{

if(centerChild.InnerText.Replace("&nbsp;", " ").Trim()!=check ||count==0)

{

check= centerChild.InnerText.Replace("&nbsp;", " ").Trim();

count++;

}

}

if (count == 1)

{

reportTable.Rows.Add(check);

}

}

/\*Extracting the informtion from the Font tag to the DataTable\*/

private void Get\_FontTagToDataTable(DataTable reportTable, HtmlNode font)

{

List<HtmlNode> fonts = font.ChildNodes.Where(x => x.NodeType == HtmlNodeType.Element && x.ParentNode.Name == font.Name).ToList();

foreach (HtmlNode item in fonts)

{

if(item.Name!="br")

Get\_ItemToDataTable(reportTable, item);

}

}

/\*Puts the DataTable in to a CSV file\*/

public void DataTableToCSV(DataTable reportTable, string file\_Name)

{

if (reportTable != null && reportTable.Rows.Count > 0)

{

// create object for the StringBuilder class

StringBuilder sb = new StringBuilder();

string strLocation = CreateDir(file\_Name);

// Get name of columns from datatable and assigned to the string array

string[] columnNames = reportTable.Columns.Cast<DataColumn>().Select(column => column.ColumnName).ToArray();

// Create comma sprated column name based on the items contains string array columnNames

sb.AppendLine(string.Join(",", columnNames));

// Fatch rows from datatable and append values as comma saprated to the object of StringBuilder class

foreach (DataRow row in reportTable.Rows)

{

IEnumerable<string> fields = row.ItemArray.Select(field => string.Concat("\"", field.ToString().Replace("\"", "\"\""), "\""));

sb.AppendLine(string.Join(",", fields));

}

// save the file

if (!File.Exists(strLocation+"\\"+ file\_Name))

{ // Create a file to write to

File.WriteAllText(strLocation, sb.ToString());

}

}

}

/\*Creating the folder for the CSV file\*/

public string CreateDir(string file\_Name)

{

StringBuilder sb = new StringBuilder();

string subDir = DateTime.Now.ToString("yyyyMMddHHmmss");

ConfigurtionJson location = this.TermConfigurtion.FirstOrDefault(tc => tc.NodeType == "file");

if (!Directory.Exists(location.TagValue))

{

Directory.CreateDirectory(location.TagValue);

}

// Create a sub directory

if (!Directory.Exists(location.TagValue + "\\" + subDir) && Program.createDir)

{

Directory.CreateDirectory(location.TagValue + "\\" + subDir);

Program.dir = location.TagValue + "\\" + subDir;

Program.createDir = false;

}

return $@"{Program.dir}\{file\_Name}.csv";

}

/\*Checking if the term from the Configuration file exist\*/

public bool IsTermConfigurtionTrue(string tagName,string tagVal=null)

{

if (tagVal == null)

{

if (this.TermConfigurtion.Find(tc => tc.TagName == tagName) != null)

return true;

return false;

}

else

{

if (this.TermConfigurtion.Find(tc => tc.TagName == tagName && tc.TagValue == tagVal) != null)

return true;

return false;

}

}

/\*NO Arguments Constructor for DataTable\*/

private DataTable Get\_HdrTable\_Report()

{

DataTable table = new DataTable("Table\_Report");

table.Columns.Add("Key", typeof(string));

table.Columns.Add("Value", typeof(string));

table.Columns.Add("timeStemp", typeof(DateTime));

table.Columns.Add("Test", typeof(string));

return table;

}

/\*NO Arguments Constructor for DataTable\*/

private DataTable Get\_SpanTable\_Report()

{

DataTable table = new DataTable("Table\_Report");

table.Columns.Add("Test Name", typeof(string));

table.Columns.Add("Received", typeof(string));

table.Columns.Add("Lower Limit", typeof(string));

table.Columns.Add("Upper Limit", typeof(string));

table.Columns.Add("Units", typeof(string));

table.Columns.Add("Status", typeof(string));

table.Columns.Add("timeStemp", typeof(DateTime));

table.Columns.Add("Test", typeof(string));

return table;

}

/\*Fix wrong string in one of the logs\*/

private string FixString(string[] withoutSpace)

{

string withoutSpace2 = "";

for (int i = 0; i < withoutSpace.Length; i++)

{

withoutSpace2 = withoutSpace2 + withoutSpace[i] + " ";

}

return withoutSpace2;

}

}

}

מחלקת **Machine**:

using System;

using System.Collections.Generic;

using System.Text;

namespace Log\_Parser\_App

{

class Machine

{

public string Company\_Code { get; set; }

public string Machine\_Code { get; set; }

public string IP\_Adress { get; set; }

public string Log\_Folder { get; set; }

public string Username { get; set; }

public string Password { get; set; }

public string Domain { get; set; }

public string Parser\_Template { get; set; }

public List<Log\_File> Files { get; set; }

}

class Log\_File

{

public string File\_Name { get; set; }

public string File\_Content { get; set; }

}

}

מחלקת **ConfigurtionJson**:

using System.Collections.Generic;

using System.IO;

using Newtonsoft.Json;

namespace Log\_Parser\_App

{

/\*Class for using the Configurtion file while the program is running\*/

class ConfigurtionJson

{

public string NodeType { get; set; }

public string TagName { get; set; }

public string TagValue { get; set; }

/\*NO Arguments Constructor\*/

public ConfigurtionJson()

{

this.NodeType = null;

this.TagName = null;

this.TagValue = null;

}

/\*2 Arguments Constructor\*/

public ConfigurtionJson(string nodeType, string TagName)

{

this.NodeType = nodeType;

this.TagName = TagName;

this.TagValue = null;

}

/\*3 Arguments Constructor\*/

public ConfigurtionJson(string nodeType, string TagName, string TagValue)

{

this.NodeType = nodeType;

this.TagName = TagName;

this.TagValue = TagValue;

}

/\*Function that read the Configurtion file and creating the object\*/

public static List<ConfigurtionJson> EnterConfigFileToList()

{

using (StreamReader jsonFile = File.OpenText(ConnectionStringConfigFile()))

{

if (jsonFile != null)

{

JsonSerializer serializer = new JsonSerializer();

List<ConfigurtionJson> listOfConfigurtionJson = (List<ConfigurtionJson>)serializer.Deserialize(jsonFile, typeof(List<ConfigurtionJson>));

return listOfConfigurtionJson;

}

else

return null;

}

}

/\*Function that return the Configurtion file path for security\*/

private static string ConnectionStringConfigFile()

{

return @"D:\Final Project\IIOT\_Collector\Log\_Parser\_App\JSONconfig.json";

}

}

}