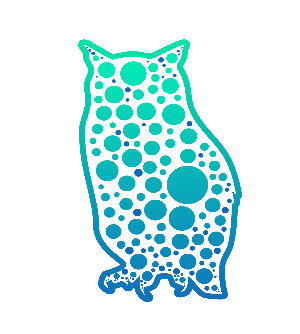
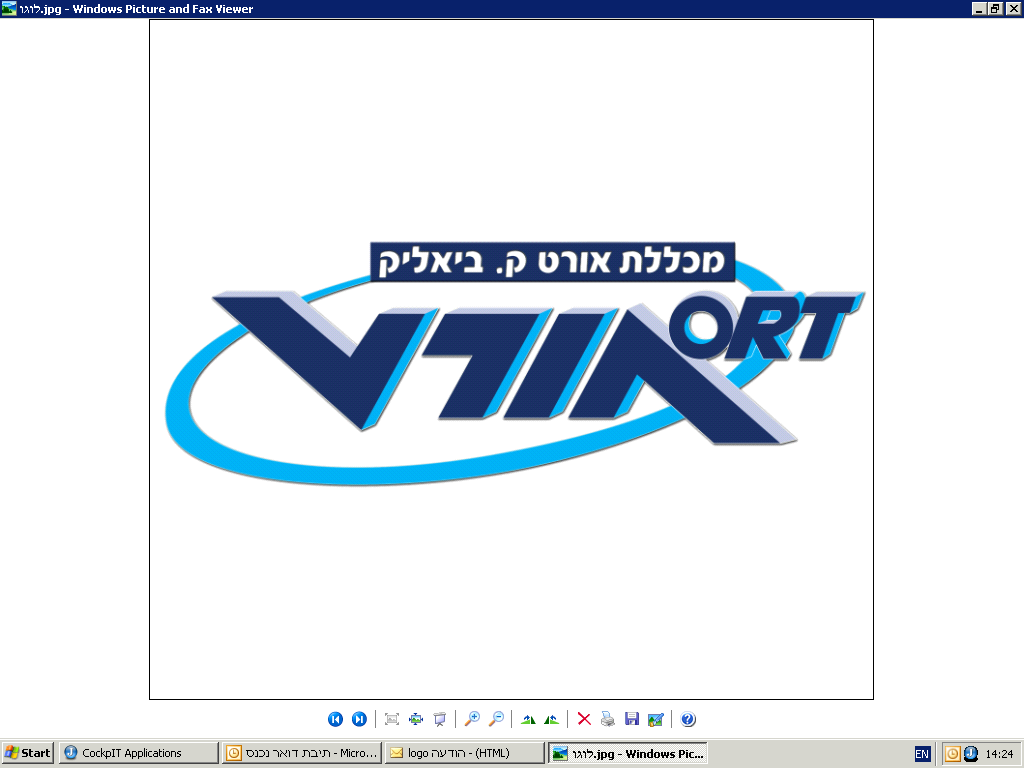
****

****

**פרויקט גמר**

-פרוייקטנט ממר''ם-

ה נ ד ס א י

ה נ ד ס ת - ת ו כ נ ה

בהתמחות: מ ח ש ב י ם

**נושא:** "פתיחת ציר" וניטור מערכת NetApp E-Series

**מגיש:** ארד גונן

**תעודת זהות:** 206394579

שנה"ל תשע"ח 2018

תוכן

[*מטרת המערכת:* 2](#_Toc513626309)

[ציוד חומרה ותוכנה הנדרשים לביצוע הפרויקט 3](#_Toc513626310)

[תהליכים עיקריים: 4](#_Toc513626311)

[כלים תוכנתיים ששולבו בפרוייקט: 5](#_Toc513626312)

[מפת המערכת: 6](#_Toc513626313)

[שלבים בפיתוח הפרוייקט: 7](#_Toc513626314)

[הפעלת המערכת: 11](#_Toc513626315)

[מבנה שרת הREST: 14](#_Toc513626316)

[*מבנה* השרת הראשי*:* 15](#_Toc513626317)

[מבנה צד לקוח: 22](#_Toc513626318)

[Admin Page Component 23](#_Toc513626319)

[Chart js Component 28](#_Toc513626320)

[Form Component 34](#_Toc513626321)

[NavBar Component 36](#_Toc513626322)

[Ngbd-modal-basic Component: 38](#_Toc513626323)

[App Component: 40](#_Toc513626324)

[Services: 43](#_Toc513626325)

# מטרת המערכת:

מערכת ניטור ופתיחת ציר עבור מכונת אחסון מסוג E-Series (NETAPP). מערכת זאת באה לפתור בעיה של חוסר יכולת ביצוע פתיחת ציר במכונות אלו, ובנוסף ומאפשרת ניטור קל בעזרת ממשק גרפי נעים לשימוש.

פתיחת ציר הוא תהליך צבאי בו מציגים תמונת מצב של כל התקלות במכונות הרלוונטיות בפתיחה. במכונות כמו ה E-Series  תהליך זה הוא ידני לגמרי ולוקח מספר שעות. בעזרת מערכת זו אני שואף לקצר את זמן פתיחת הציר לדקות בודדות.

מערכת זו אוספת נתונים מהלוגים של המכונה, מנתחת אותם ומציגה בצורה גרפית תקלות חומרה וניהול של המכונה.

מטרת המערכת, היא "פתיחת ציר" – מושג המתאר תהליך של חיבור בין שני שרתים והעברת מידע ביניהם. הצורך במערכת כזו נובע מכך שהמערכת E-Series של חברת NetApp לא נותנת אפשרות מובנת להצגת הודעות ממערכת האחסון, אלא רק קבצי לוג שאותם צריך לנתח. המערכת פותחה במטרה לתת דרך קלה ומהירה לבדוק את תקינותה של המערכת, המשמשת את היחידה בצורה נרחבת. המערכת נבנתה בסביבת Web (אזכיר שלא נלמד פיתוח בסביבה זו במכללה, אלא אני למדתי לבד כיצד לפתח בה). השרת המריץ את המערכת מבוסס Node.js שזוהי מעטפת TypeScript(JavaScript) המשמשת לתכנות צד שרת. צד הלקוח, מבוסס Angular 5, (הדרך המודרנית ביותר כיום לפיתוח Web), העיצוב מבוסס Bootstrap, Angular-Material,ng-bootstrap. מסד הנתונים המשמש אותי הוא MongoDB , מכיוון שיזרום מידע רב דרך המערכת ולכן צריך מסד המיועד לשימוש Big Data.

# ציוד חומרה ותוכנה הנדרשים לביצוע הפרויקט

מבני נתונים- רשימות וטבלאות.

קבצים- קבצי קונפיגורציה (יכילו מידע על המערכות שצריך להתחבר אליהם).

בסיס נתונים-מבוסס MongoDB , ישמור את תוצאות פתיחת הציר האחרונה על מנת לזרז גישת משתמשים.

אלגוריתמים-ניתוח קבצי JSON  שנקבל אותם מהמערכת - בעזרת REST api  .

תכנות רשת- חיבור מרחוק בעזרת HTTP  וRESTAPI לשרת הניהול של המכונה. חיבור לדומיין לצורך זיהוי משתמש בכניסתו, בעזרת ntlm-express

עמדת פיתוח - Windows  עם NodeJs וMongoDB. וAngularCli

עמדת משתמש - כל עמדה עם Chrome

# תהליכים עיקריים:

-התחברות לשרת RESTapi לשם קבלת מידע מהמערכות.

- התחברות למסד נתונים MongoDB

-לכל לקוח שמתחבר, המערכת תנסה לקבל את שם המשתמש המחובר לwindows, במידה ולא תצליח הוא ייקרא Guest .

- המערכת תקבל רשימה של כל התקלות במערכת האחסון ותייצר טבלאות המייצגות כל מערכת ומערכת ברשת.

* התחברות למערכת – התחברות בצורת SSO
* צפייה במצבן הכללי של כל המכונות – חלון ה Dashboard
* צפייה בתקלות חומרה בקטגוריה מסויימת – (Disks , Mgmt , Hardware , FC , Pools&Vols)
* ייצוא של פרטי התקלה והרכיב התקול והמערכת אליה הוא שייך בCSV וHTML
* צפייה בתקלות השוטפות במכונות
* עדכון הנתונים – בצורה ידנית (פתיחת ציר עלולה לקחת מספר דקות – כדי למנוע מצב של כניסה למערכת ולא מוצג שום מידע

# כלים תוכנתיים ששולבו בפרוייקט:

כל הפרוייקט מבוסס בבסיסו על JavaScript + HTML. צד השרת מבוסס מעטפת JavaScript בשם Node.js - זוהי מעטפת מאוד יעילה לתכנות צד שרת בזכות זה שהיא אסינכרונית כלומר אין מצב שתהיה שורה שלוקח לה הרבה זמן להתבצע כי במקום להמתין הקומפיילר יעבור לשורה הבאה לביצוע.

צד הלקוח מבוסס Angular 5 - בנוסף השתמשתי בכמה מודולים קיימים המוסיפים פונקציונליות משופרת.

Charts.js - מודול המציג דיאגרמות, איתו כלול pieceLabel.js המאפשר לכתוב על הדיאגרמות.

Angular Material – מודול לעיצוב של גוגל לAngular.

Ng-bootstrap - מודול עיצוב בוטסטראפ לממשק Angular של צד לקוח

Ngx-json-viewer – מודול להצגה יפה ונוחה של קבצי JSON

Ntlm-express – מודול להטמעת sso - כניסה חד פעמית למערכת, המודול מזהה את המשתמש המחובר לווינדוס ומעביר למערכת את שם המשתמש

Express - מודול לשרת

במהלך הפרוייקט עברתי לשימוש עם כלי לניהול גרסאות Git, על מנת לחסוך סיבוך של גרסאות כי עבדתי בכמה מקומות ומחשבים שונים. לכן השתמשתי בBitBucket כשרת, וsourceTree ככלי משתמש.

# מפת המערכת:

# שלבים בפיתוח הפרוייקט:

*עד סוף נובמבר 2017 – סיום תהליך הוצאת תקלות מכל קטגוריה וייצוא המידע.*

*עד סוף 2017 – שאיבת הנתונים מהמכונות ב RestAPI  ותחילת עבודה על MongoDB.*

*עד סוף ינואר 2018 – הרצת \פתיחת ציר על פי לוז ואחסון התוצאות בMongoDB.*

*עד סוף פברואר 2018 – עבודה על מנגנון SSO.*

*מרץ 2018 – ליטושים \ תיקוני באגים.*

*עד סוף מרץ 2018 –  מערכת עובדת 90-100% מהתכנון המקורי.*

היסטוריית Commit: שרת + לקוח

| **Author** | **Commit** | **Message** | **Date** |
| --- | --- | --- | --- |
| [https://bitbucket.org/account/aradgonen/avatar/16/?ts=1525845806](https://bitbucket.org/aradgonen/)Arad Gonen | [c875c02](https://bitbucket.org/aradgonen/eseriesproject/commits/c875c0221dd857960577c7ce5c89dba071f52812) | update modules and fix tooltip | 13 hours ago |
| [https://bitbucket.org/account/aradgonen/avatar/16/?ts=1525845806](https://bitbucket.org/aradgonen/)Arad Gonen | [3a4749c](https://bitbucket.org/aradgonen/eseriesproject/commits/3a4749c9f0423fb17cab68a77dd4d5b6726e5471) | tooltip | 16 hours ago |
| [https://bitbucket.org/account/aradgonen/avatar/16/?ts=1525845806](https://bitbucket.org/aradgonen/)Arad Gonen | [9ac380b](https://bitbucket.org/aradgonen/eseriesproject/commits/9ac380b33951baf1135dff915996fffa450cc05e) | tooltip | 19 hours ago |
| [https://bitbucket.org/account/aradgonen/avatar/16/?ts=1525845806](https://bitbucket.org/aradgonen/)Arad Gonen | [32a6b4a](https://bitbucket.org/aradgonen/eseriesproject/commits/32a6b4a075271a2a40c0bc841c64d1aeabc23a05) | update - added question mark for help | 2 days ago |
| [https://bitbucket.org/account/aradgonen/avatar/16/?ts=1525845806](https://bitbucket.org/aradgonen/)Arad Gonen | [f09ceee](https://bitbucket.org/aradgonen/eseriesproject/commits/f09ceeef1f93c3de54c8491e5af350071e4aa236) | update | 2 days ago |
| [https://bitbucket.org/account/aradgonen/avatar/16/?ts=1525845806](https://bitbucket.org/aradgonen/)Arad Gonen | [a1529eb](https://bitbucket.org/aradgonen/eseriesproject/commits/a1529eb93c7d4372d636cdfd004931758f1addc5) | #2 upload node\_modules files | 2 days ago |
| [https://bitbucket.org/account/aradgonen/avatar/16/?ts=1525845806](https://bitbucket.org/aradgonen/)Arad Gonen | [905028a](https://bitbucket.org/aradgonen/eseriesproject/commits/905028a3e4d02474d305b07d443759785140737c) | #1 commit to new file in folder node\_modules | 2 days ago |
| [https://bitbucket.org/account/aradgonen/avatar/16/?ts=1525845806](https://bitbucket.org/aradgonen/)Arad Gonen | [2a7e3eb](https://bitbucket.org/aradgonen/eseriesproject/commits/2a7e3eb8f350dc441c9b0388eb7d172d696b2892) | .gitignore edited online with Bitbucket | 2 days ago |
| [https://bitbucket.org/account/aradgonen/avatar/16/?ts=1525845806](https://bitbucket.org/aradgonen/)Arad Gonen | [2585099](https://bitbucket.org/aradgonen/eseriesproject/commits/2585099a2b3d18b43d050c2683533c1834a2f5c5) | fix sso | 2 days ago |
| [https://bitbucket.org/account/aradgonen/avatar/16/?ts=1525845806](https://bitbucket.org/aradgonen/)Arad Gonen | [0b84ba7](https://bitbucket.org/aradgonen/eseriesproject/commits/0b84ba75283bc019ff7d021a1618f8685e169d96) | cleaning | 2 days ago |
| [https://bitbucket.org/account/aradgonen/avatar/16/?ts=1525845806](https://bitbucket.org/aradgonen/)Arad Gonen | [a638ac1](https://bitbucket.org/aradgonen/eseriesproject/commits/a638ac11a68f35ff9dac85add1dcd9dca7f6b3f7) | fix | 2 days ago |
| [https://bitbucket.org/account/aradgonen/avatar/16/?ts=1525845806](https://bitbucket.org/aradgonen/)Arad Gonen | [98ec58d](https://bitbucket.org/aradgonen/eseriesproject/commits/98ec58d848414f7f4e4f9813c22b33fcdf21c20a) | preparing ngb modal component to new data and new failures data | 3 days ago |
| [https://bitbucket.org/account/aradgonen/avatar/16/?ts=1525845806](https://bitbucket.org/aradgonen/)Arad Gonen | [cef4b5b](https://bitbucket.org/aradgonen/eseriesproject/commits/cef4b5b3919d39e38c4db95b799540786c1dd2e7) | update to project document | 2018-05-02 |
| [https://bitbucket.org/account/aradgonen/avatar/16/?ts=1525845806](https://bitbucket.org/aradgonen/)Arad Gonen | [1a3dca6](https://bitbucket.org/aradgonen/eseriesproject/commits/1a3dca6a92317dfa4387c9fe98e53cd8d56364c6) | fix all expand - to expand on request | 2018-05-02 |
| [https://bitbucket.org/account/aradgonen/avatar/16/?ts=1525845806](https://bitbucket.org/aradgonen/)Arad Gonen | [2b9fd0a](https://bitbucket.org/aradgonen/eseriesproject/commits/2b9fd0a3b9e3e079c74123e2e943b440e88db861) | Update | 2018-05-02 |
| [https://bitbucket.org/account/aradgonen/avatar/16/?ts=1525845806](https://bitbucket.org/aradgonen/)Arad Gonen | [4491409](https://bitbucket.org/aradgonen/eseriesproject/commits/4491409373f5780bd0ae4f196581bb8e18dd65a0) | fix design | 2018-05-01 |
| [https://bitbucket.org/account/aradgonen/avatar/16/?ts=1525845806](https://bitbucket.org/aradgonen/)Arad Gonen | [8791eff](https://bitbucket.org/aradgonen/eseriesproject/commits/8791efffe4a6e2c4518c8283f11a779ccabd7380) | added module | 2018-05-01 |
| [https://bitbucket.org/account/aradgonen/avatar/16/?ts=1525845806](https://bitbucket.org/aradgonen/)Arad Gonen | [093e300](https://bitbucket.org/aradgonen/eseriesproject/commits/093e3001561883dda68dc59faeb8fd0bf3367962) | added module | 2018-05-01 |
| [https://bitbucket.org/account/aradgonen/avatar/16/?ts=1525845806](https://bitbucket.org/aradgonen/)Arad Gonen | [388d593](https://bitbucket.org/aradgonen/eseriesproject/commits/388d59351ff251fc59bacfb76f3950c198479b9c) | fix- set variables for json viewer module | 2018-05-01 |
| [https://bitbucket.org/account/aradgonen/avatar/16/?ts=1525845806](https://bitbucket.org/aradgonen/)Arad Gonen | [c794902](https://bitbucket.org/aradgonen/eseriesproject/commits/c79490299760182bfe6fcef8777013840b76ec88) | fix some bugs | 2018-05-01 |
| [https://bitbucket.org/account/aradgonen/avatar/16/?ts=1525845806](https://bitbucket.org/aradgonen/)Arad Gonen | [e2dd752](https://bitbucket.org/aradgonen/eseriesproject/commits/e2dd752072535d91cc88f78022a1487a10456897) | added the documentation | 2018-05-01 |
| [https://bitbucket.org/account/aradgonen/avatar/16/?ts=1525845806](https://bitbucket.org/aradgonen/)Arad Gonen | [283dfa8](https://bitbucket.org/aradgonen/eseriesproject/commits/283dfa8594582e162093fef5abbe064765964fc0) | fix - problem with backslashes in admin authentication | 2018-04-29 |
| [https://bitbucket.org/account/aradgonen/avatar/16/?ts=1525845806](https://bitbucket.org/aradgonen/)Arad Gonen | [a49c3ac](https://bitbucket.org/aradgonen/eseriesproject/commits/a49c3acdd6bdec0c9bcb0d1eadff6875c505bd8d) | fix express missing | 2018-04-25 |
| [https://bitbucket.org/account/aradgonen/avatar/16/?ts=1525845806](https://bitbucket.org/aradgonen/)Arad Gonen | [9df9e4c](https://bitbucket.org/aradgonen/eseriesproject/commits/9df9e4ceb09c3355f31e0bc43e8c82da5a70348e) | fix pools from pol | 2018-04-25 |
| [https://bitbucket.org/account/aradgonen/avatar/16/?ts=1525845806](https://bitbucket.org/aradgonen/)Arad Gonen | [ea7e443](https://bitbucket.org/aradgonen/eseriesproject/commits/ea7e4438bb03d4aa846e1ab32ccf52534a3b94b3) | remove cancel button from global modal | 2018-04-25 |
| [https://bitbucket.org/account/aradgonen/avatar/16/?ts=1525845806](https://bitbucket.org/aradgonen/)Arad Gonen | [b115c6b](https://bitbucket.org/aradgonen/eseriesproject/commits/b115c6bfc058f8aabdb31b3e9acb24f2789937aa) | fix navbar main active color | 2018-04-25 |
| [https://bitbucket.org/account/aradgonen/avatar/16/?ts=1525845806](https://bitbucket.org/aradgonen/)Arad Gonen | [920dbe7](https://bitbucket.org/aradgonen/eseriesproject/commits/920dbe7b15ab55e5d101511338cacf479413e8af) | fix brand alignment | 2018-04-25 |
| [https://bitbucket.org/account/aradgonen/avatar/16/?ts=1525845806](https://bitbucket.org/aradgonen/)Arad Gonen | [f7e7eb5](https://bitbucket.org/aradgonen/eseriesproject/commits/f7e7eb5f1d3065bc4d570b692a63322c68b1a245) | code cleanup and some ui alignment | 2018-04-25 |
| [https://bitbucket.org/account/aradgonen/avatar/16/?ts=1525845806](https://bitbucket.org/aradgonen/)Arad Gonen | [ac6ad28](https://bitbucket.org/aradgonen/eseriesproject/commits/ac6ad28be26b3958302aa2e0c57e7411e54247cc) | Card background for pies | 2018-04-25 |
| [https://bitbucket.org/account/aradgonen/avatar/16/?ts=1525845806](https://bitbucket.org/aradgonen/)Arad Gonen | [87a92e1](https://bitbucket.org/aradgonen/eseriesproject/commits/87a92e190430c0bb0c39959bdf2991c0fb03fdb8) | change chart name from id to name |  |
|  |
| [https://bitbucket.org/account/aradgonen/avatar/16/?ts=1525845806](https://bitbucket.org/aradgonen/)Arad Gonen | [562be45](https://bitbucket.org/aradgonen/eseriesproject/commits/562be45c52a1506702ca39f41072933852a9096a) | new logo |
| [https://bitbucket.org/account/aradgonen/avatar/16/?ts=1525845806](https://bitbucket.org/aradgonen/)Arad Gonen | [ec14eeb](https://bitbucket.org/aradgonen/eseriesproject/commits/ec14eebbc34712b7e8e4f2fdd3e2374d66957d06) | First Commit - starting to work with git |
| [https://bitbucket.org/account/aradgonen/avatar/16/?ts=1525845806](https://bitbucket.org/aradgonen/)Arad Gonen | [e60b107](https://bitbucket.org/aradgonen/eseriesproject/commits/e60b1070d3b1fc8de850973c354132f51d3380c4) | Initial commit |

שרת דמה:

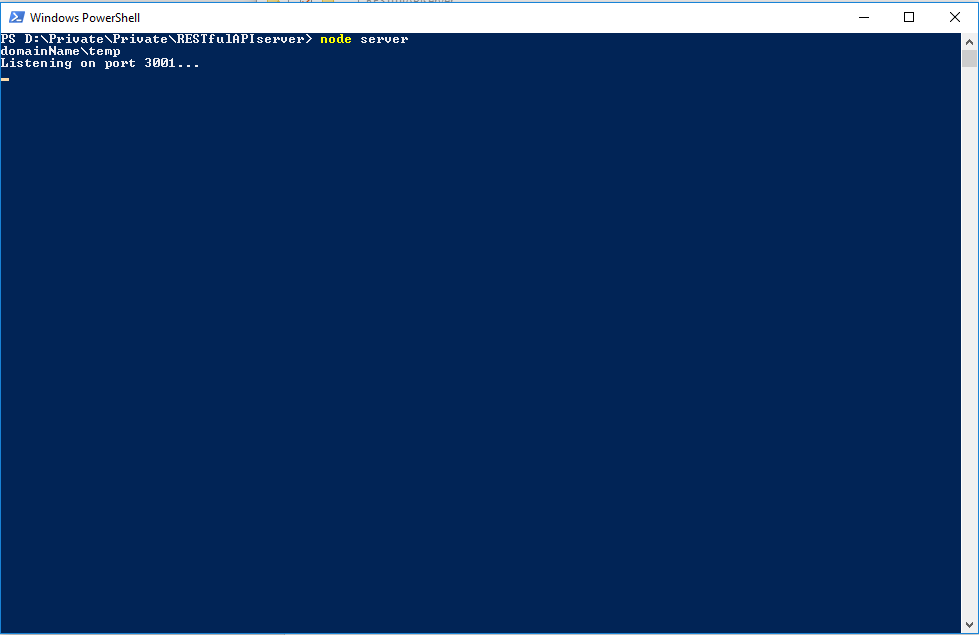
| **Author** | **Commit** | **Message** | **Date** |
| --- | --- | --- | --- |
| [https://bitbucket.org/account/aradgonen/avatar/16/?ts=1525845806](https://bitbucket.org/aradgonen/)Arad Gonen | [c6f3c86](https://bitbucket.org/aradgonen/restapidemo/commits/c6f3c86caeefc58f1f917fa763eaddba97edbf7c) | fix | 2 days ago |
| [https://bitbucket.org/account/aradgonen/avatar/16/?ts=1525845806](https://bitbucket.org/aradgonen/)Arad Gonen | [35918ac](https://bitbucket.org/aradgonen/restapidemo/commits/35918ac40f1572a10d35ccad9cdd8329c723e19f) | added picture to use in project documentation and code comments | 2018-05-02 |
| [https://bitbucket.org/account/aradgonen/avatar/16/?ts=1525845806](https://bitbucket.org/aradgonen/)Arad Gonen | [8e322b8](https://bitbucket.org/aradgonen/restapidemo/commits/8e322b870246a4d740e8ecda48ea6ebec9e0735f) | Added Powershell script file to quickly run the server | 2018-04-29 |
| [https://bitbucket.org/account/aradgonen/avatar/16/?ts=1525845806](https://bitbucket.org/aradgonen/)Arad Gonen | [c73dd82](https://bitbucket.org/aradgonen/restapidemo/commits/c73dd8252072caed24a30fc7a6b75f44de8a1091) | First Upload | 2018-04-29 |
| [https://bitbucket.org/account/aradgonen/avatar/16/?ts=1525845806](https://bitbucket.org/aradgonen/)Arad Gonen | [b8927a7](https://bitbucket.org/aradgonen/restapidemo/commits/b8927a783a5ba7321805d5ff38d43288d6cef1bd) | Initial commit |  |
|  |  |  |  |
|  |  |  |  |

# הפעלת המערכת:

-קודם כל בגלל שזה פרוייקט צבאי שנועד לפעול בממר''ם, נצטרך להפעיל שרת דמה הידמה את פעולת מכונת האחסון ושיתמוך בrestapi.

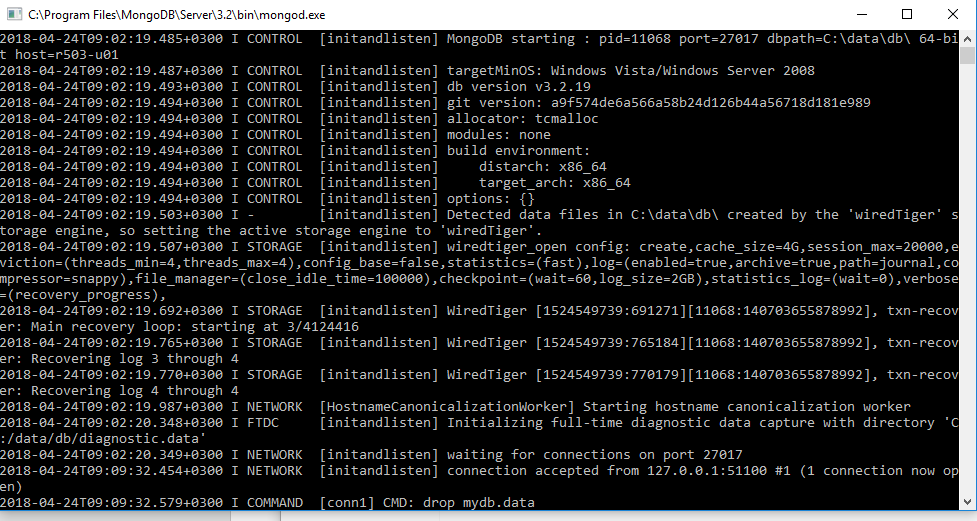
כך נפעיל את השרת:

נפתח חלון powershell בתיקייה בה נמצאים קבצי השרת -> נכתוב את הפקודה node server



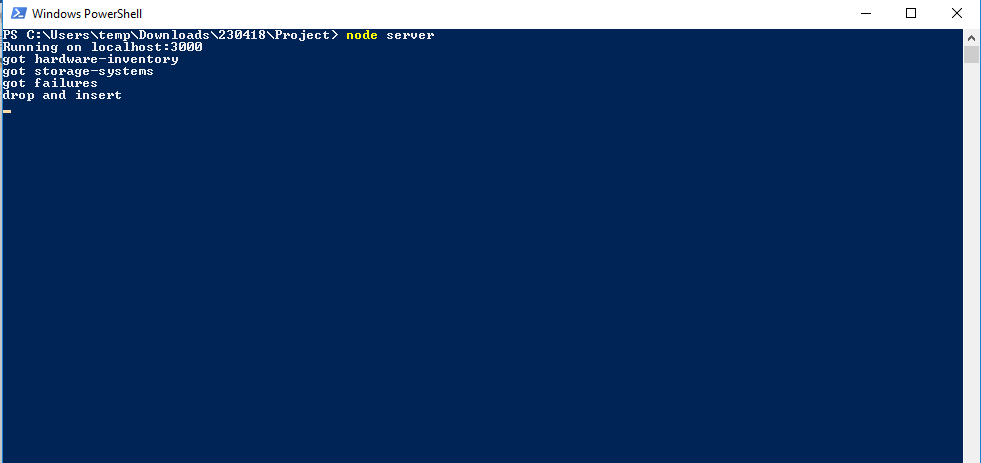
לאחר מכן, נצטרך להפעיל את מסד הנתונים (במקרה שהוא לא מותקן כservice)

נכנס לכונן C: -> לתקייה program files - > לתיקייה MongoDB -> לתקייה בשם server - > לתקייה בשם 3.2 -> לתקייה bin - > נריץ קובץ בשם mongod



ועכשיו ניתן להריץ את השרת הראשי

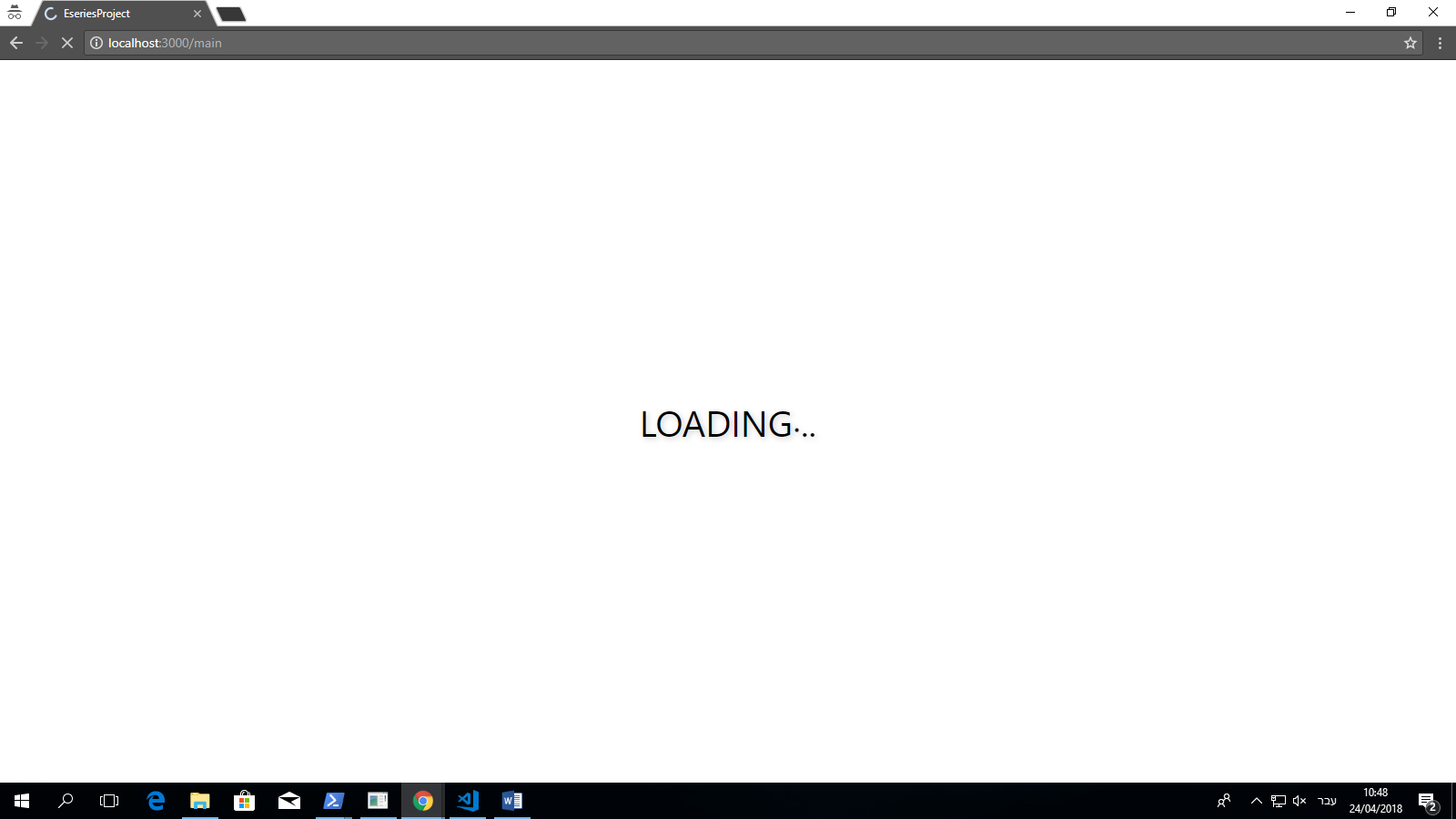
ניכנס לתיקייה של הפרויקט, נפתח חלון powershell ונכתוב את הפקודה node server (במידה ונעשה שינוי במערכת נריץ לפני את פקודה ng build).



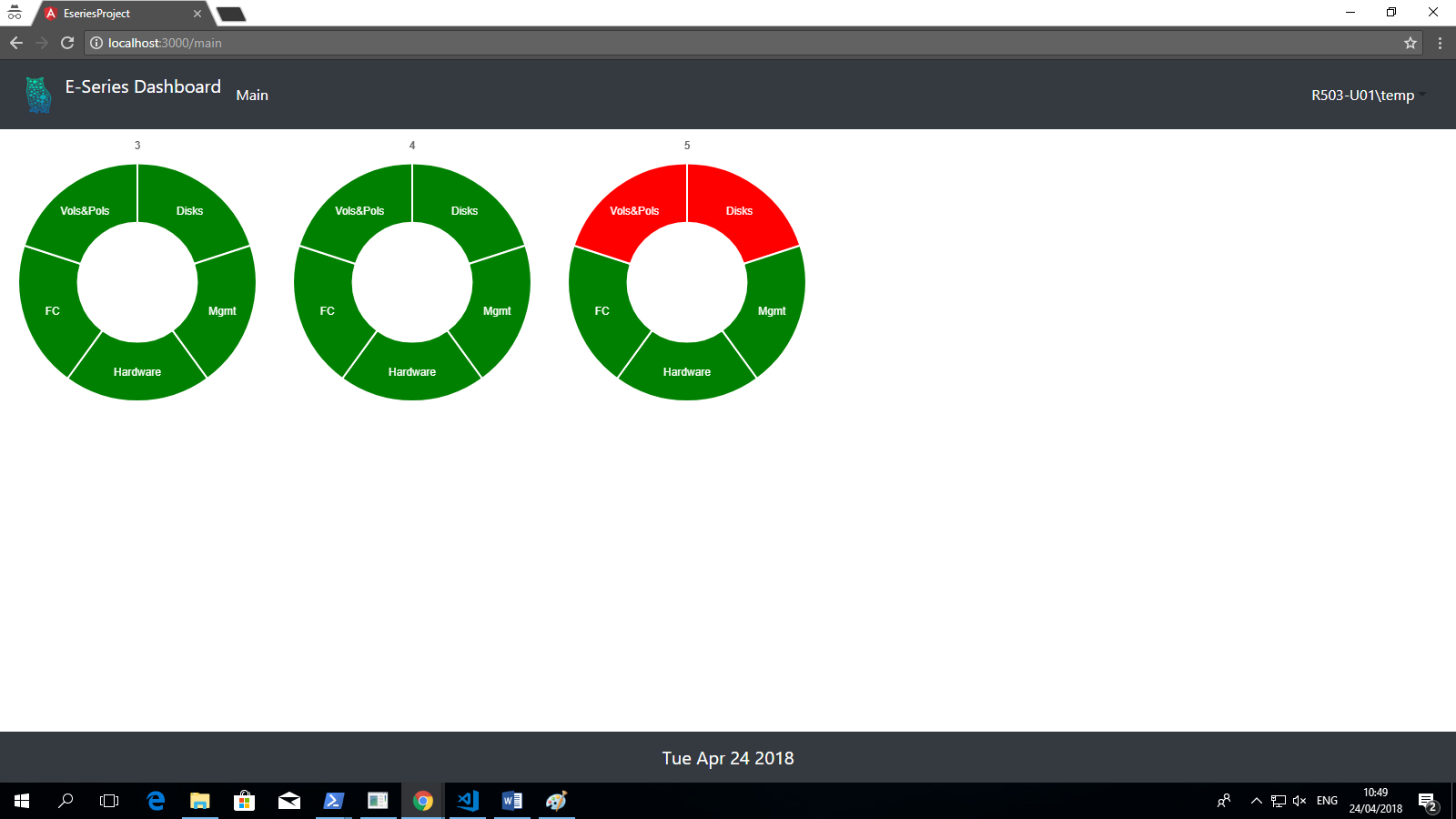
עכשיו ניכנס לדפדפן (כרום) ונכתוב: אם אנחנו במחשב שבו רץ השרת, localhost:3000

במידה ואנחנו על מחשב אחר ברשת, נכתוב את כתובת הip של השרת לדוגמא 192.168.1.123:3000

בהתחלה נראה מסך טעינה:



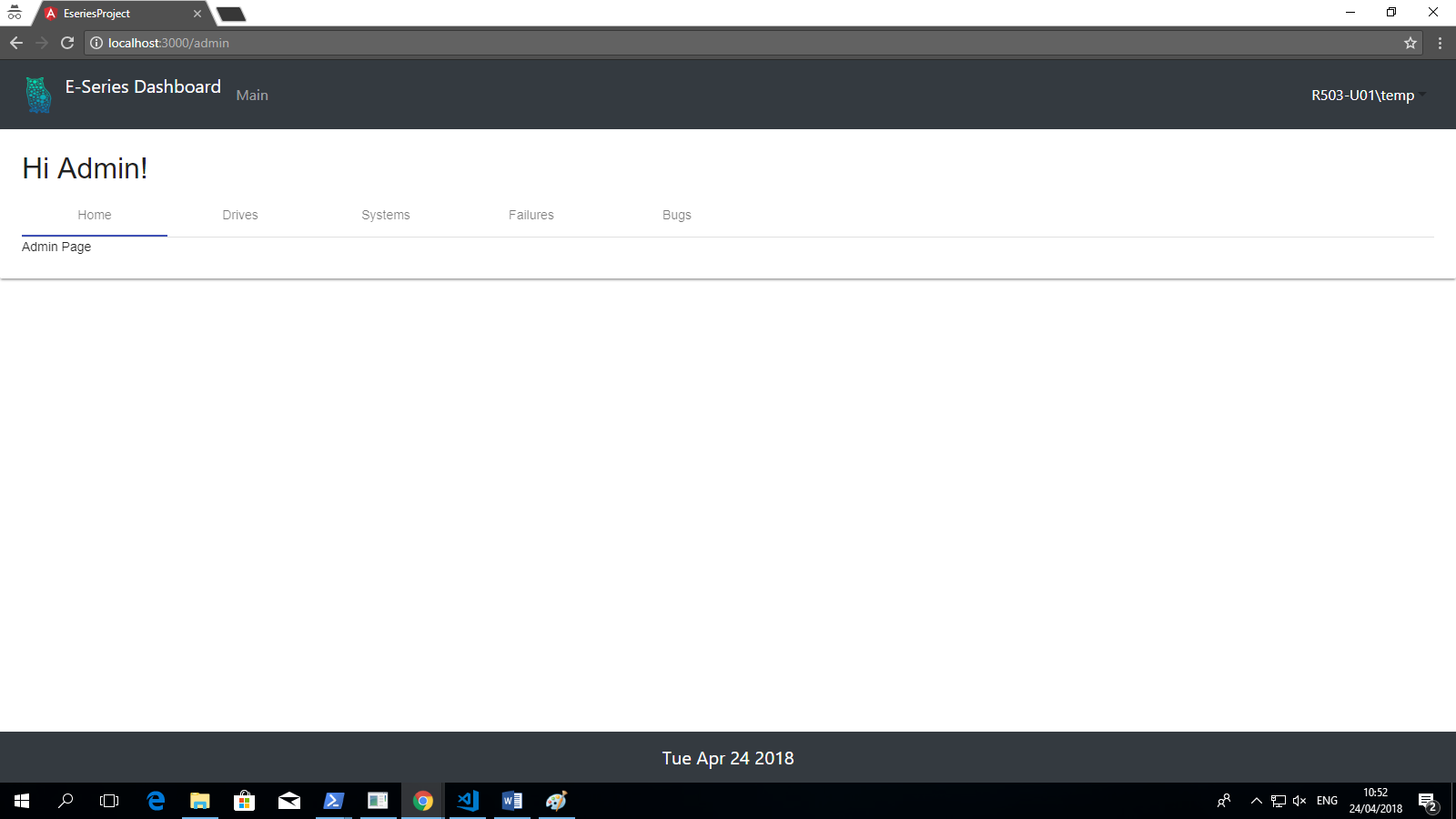
ואז יעלה המסך הראשי:



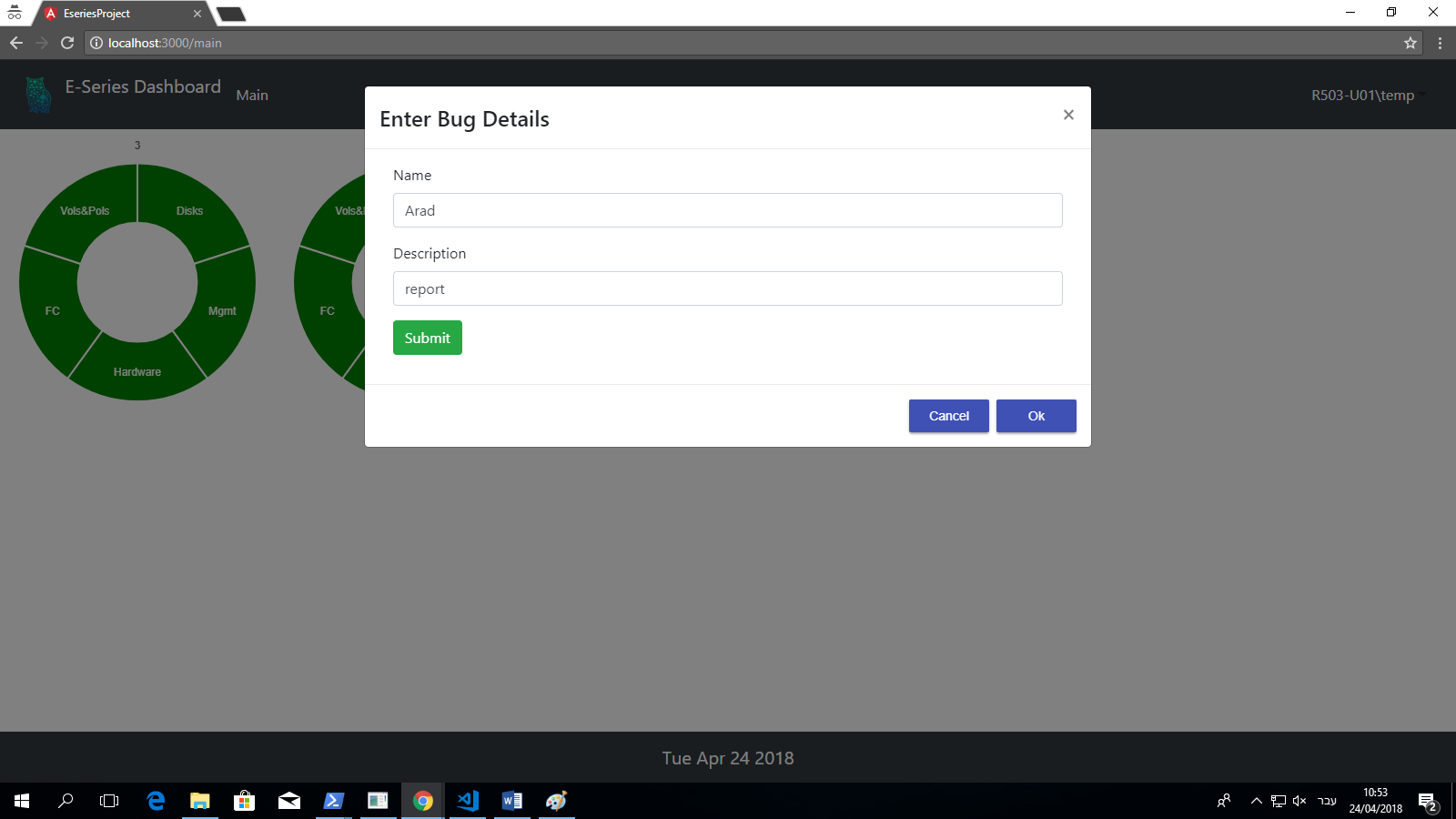
אם נלחץ על שם המשתמש, ייפתח תפריט משתמש – במידה והמשתמש המחובר מוגדר במערכת כמנהל, יופיע לו האפשרות להגיע לדף הניהול של המערכת



דף מנהל המערכת נראה כך:



במידה ונרצה לשלוח הודעה על באג, נלחץ על bug report ונקבל מסך כזה:



# מבנה שרת הREST:

* + בתיקייה Arad נמצאים קבצי log זמניים שקיבלתי מממר''ם
  + בתיקייה node\_modules נמצאים קבצי הגדרה של node.js
  + בתיקייה routes קיים קובץ routes.js

שהוא הקובץ שבו נמצא הקוד העונה לhttp request

* + Package.json קובץ הגדרה של node.js
  + Server.js קובץ השרת

Server.js:

**var** express **=** require**(**'express'**);**//express module

**var** app **=** express**();**//assign the app variable to express

**var** fs **=** require**(**'fs'**);**//import the filesystem module

**var** bodyParser **=** require**(**"body-parser"**);**//import body-parser

**var** path **=** require**(**'path'**);**//import path module

**var** userName **=** process**.**env**[**'USERPROFILE'**].**split**(**path**.**sep**)[**2**];**

**var** loginId **=** path**.**join**(**"domainName"**,**userName**);**

console**.**log**(**loginId**);**

app**.**use**(**express**.static(**'Arad'**));**//set main folder for assets

app**.**use**(**bodyParser**.**json**());**//assign to use the module

app**.**use**(**bodyParser**.**urlencoded**({** extended**:** **true** **}));**

**var** routes **=** require**(**"./routes/routes.js"**)(**app**);**//initialize the routes

**var** server **=** app**.**listen**(**3001**,** **function** **()** **{**

console**.**log**(**"Listening on port %s..."**,** server**.**address**().**port**);**

**});**//start the server on specific port

Routes.js:

**var** appRouter **=** **function(**app**)** **{**

app**.**get**(**"/storage-systems"**,** **function(**req**,** res**)** **{**//when http request like this arrive, send the file

res**.**sendFile**(**'storage-systems.json'**,** **{**root**:**'Arad'**});**

**});**

app**.**get**(**"/storage-systems/5/hardware-inventory"**,** **function(**req**,** res**)** **{**//when http request like this arrive, send the file

res**.**sendFile**(**'hardware-inventory.json'**,** **{**root**:**'Arad'**});**

**});**

app**.**get**(**"/storage-systems/5/failures"**,** **function(**req**,** res**)** **{**//when http request like this arrive, send the file

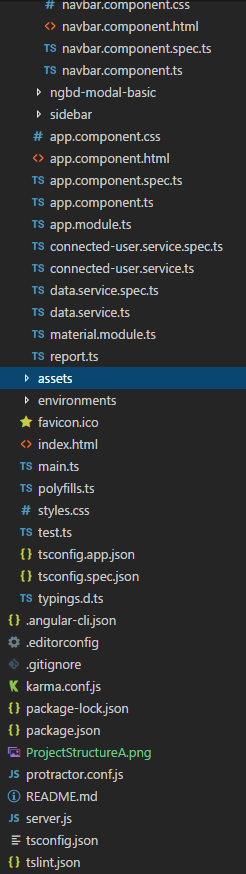
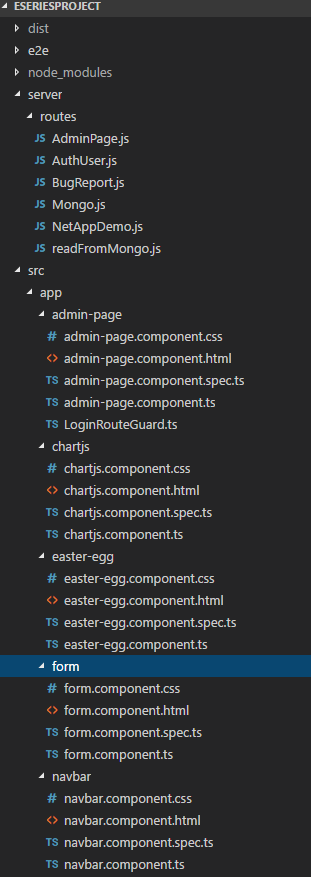
res**.**sendFile**(**'failures.json'**,** **{**root**:**'Arad'**});**

**});**

**};**

module**.**exports **=** appRouter**;**//export the router to use in other file, server file

# מבנה השרת הראשי:



תיקיית Routes:

מכילה קבצי הגדרה לשרת לטיפול בבקשות HTTP.

AdminPage.js:

AuthUser.js: יחזיר את המידע שמתקבל מהמודול node-sspi ויחזיר תגובת http עם פרטי המשתמש המחובר

BugReport.js : שומר למסד הנתונים פרטי תקלות ממשתמשים וגם מושך מהמסד כדי להציג את כל התקלות

Mongo.js: מכיל פונקציות תקשורת עם מסד הנתונים

NetAppDemo.js : מבוסס פרוטוקול RESTapi לתקשורת עם שרת הדמה, בעתיד יתקשר עם המערכת עצמה.

readFromMongo.js: מכיל פונקציות לקריאה ממסד הנתונים

Server.js: קובץ ההגדרה של השרת

קוד:

AdminPage.js:

**const** express **=** require**(**'express'**);**//express module

**const** router **=** express**.**Router**();**//express-router module

**const** path **=** require**(**'path'**);**

router**.**get**(**'/Info'**,** **(**req**,** res**)** **=>** **{**

res**.**sendFile**(**path**.**join**(**\_\_dirname**,** '../views/adminPage.html'**));**

**});**

module**.**exports **=** router**;**

AuthUser.js:

**const** express **=** require**(**'express'**);**//express module

**const** router **=** express**.**Router**();**//express-router module

**var** path **=** require**(**'path'**);**

**var** userName **=** process**.**env**[**'USERPROFILE'**].**split**(**path**.**sep**)[**2**];**

**var** loginId **=** path**.**join**(**"domainName"**,**userName**);**

**const** MongoClient **=** require**(**'mongodb'**).**MongoClient**;**//mongo module and client

**const** ObjectID **=** require**(**'mongodb'**).**ObjectID**;**//object ID-useful for data sorting

**let** response**={**

**status:** 200**,**

data**:** **[],**

message**:** **null**

**};**

**const** connection **=** **(**closure**)** **=>** **{**

**return** MongoClient**.**connect**(**'mongodb://localhost:27017/mydb'**,** **(**err**,** client**)** **=>** **{**

**if** **(**err**)** **return** console**.**log**(**err**);**

**let** db **=** client**.**db**(**'mydb'**);**

closure**(**db**);**

**});**

**};**

router**.**get**(**'/UserInfo'**,** **(**req**,** res**,**next**)** **=>** **{**

**var** nodeSSPI **=** require**(**'node-sspi'**);**

**var** nodeSSPIObj **=** **new** nodeSSPI**({**

retrieveGroups**:** **true**

**});**

nodeSSPIObj**.**authenticate**(**req**,** res**,** **function(**err**){**

res**.**finished **||** next**()**

**});**

**var** userName**=**req**.**connection**.**user**;**

**if(**userName**==**undefined**)**

**return;**

**var** userSid**=**req**.**connection**.**userSid**;**

**if(**userSid**==**undefined**)**

**return;**

**var** user **=** **[];**

**var** groups**=[];**

**if** **(**req**.**connection**.**userGroups**)** **{**

**for** **(var** i **in** req**.**connection**.**userGroups**)** **{**

**if(**req**.**connection**.**userGroups**[**i**]==**undefined**)**

**return;**

groups**.**push**(**req**.**connection**.**userGroups**[**i**]);**

**}**

**}**

user**.**push**({**UserName**:**userName**,**UserSID**:**userSid**,**userGroups**:**groups**});**

response**.**data **=** user**;**

**if(**res**.**statusCode**==**200**)**

{

console.log(res.statusCode);

res.json(response);

}

console.log(this.username);

});

router.get('/UserInfo/Admins', (req, res) => {

connection((db) => {

db.collection('users')

.find()

.toArray()

.then((users) => {

res.json(users);

})

.catch((err) => {

sendError(err, res);

});

});

});

module.exports = router;

BugReport.js :

**const** express **=** require**(**'express'**);**//express module

**const** router **=** express**.**Router**();**//express-router module

**const** MongoClient **=** require**(**'mongodb'**).**MongoClient**;**//mongo module and client

**const** ObjectID **=** require**(**'mongodb'**).**ObjectID**;**//object ID-useful for data sorting

//connect block

**const** connection **=** **(**closure**)** **=>** **{**

**return** MongoClient**.**connect**(**'mongodb://localhost:27017/mydb'**,** **(**err**,** client**)** **=>** **{**

**if** **(**err**)** **return** console**.**log**(**err**);**

**let** db **=** client**.**db**(**'mydb'**);**

closure**(**db**);**

**});**

**};**

router**.**post**(**'/SaveReport'**,** **(**req**,** res**)** **=>** **{**

**var** newReport **=** req**.**body**;**

console**.**log**(**newReport**);**

connection**((**db**)=>{**

db**.**collection**(**'BugReports'**).**insert**(**newReport**,function(**err**,**res**){**

**if(**err**)** **throw** err**;**

console**.**log**(**"New Report In DB"**);**

**});**

**})**

**});**

**let** response**={**

**status:** 200**,**

data**:** **[],**

message**:** **null**

**};**

router**.**get**(**'/AllBugs'**,** **(**req**,** res**)** **=>** **{**

connection**((**db**)** **=>** **{**

db**.**collection**(**'BugReports'**)**

**.**find**()**

**.**toArray**()**

**.**then**((**users**)** **=>** **{**

response**.**data **=** users**;**

res**.**json**(**response**);**

**})**

**.catch((**err**)** **=>** **{**

sendError**(**err**,** res**);**

**});**

**});**

**});**

module**.**exports **=** router**;**

Mongo.js:

**const** express **=** require**(**'express'**);**//express module

**const** router **=** express**.**Router**();**//express-router module

**const** MongoClient **=** require**(**'mongodb'**).**MongoClient**;**//mongo module and client

**var** dataToClient**;**

**var** url **=** 'mongodb://localhost:27017/mydb'**;**

**function** writeToDB **(**DataToDB**,**collection\_name**)**

**{**

**if(**DataToDB**!=**undefined**)**

**{**

MongoClient**.**connect**(**url**,** **function** **(**err**,** db**)** **{**

**if** **(**err**)** **throw** err**;**

db**.**collection**(**collection\_name**).**count**(function(**err**,**count**){**

**if(**count**!=**0**)**

**{**

db**.**collection**(**collection\_name**).**drop**(function(**err**,**reply**){**

**if(**err**)** **throw** err**;**

**});**

db**.**collection**(**collection\_name**).**insert**(**DataToDB**,function(**err**,**res**){**

**if(**err**)** **throw** err**;**

**});**

console**.**log**(**"drop and insert"**);**

**}**

**else**

**{**

db**.**collection**(**collection\_name**).**insert**(**DataToDB**,function(**err**,**res**){**

**if(**err**)** **throw** err**;**

**});**

console**.**log**(**"insert"**);**

**}**

**});**

**});**

**}**

**else**

**{**

console**.**log**(**"ERROR - GOT UNDEFINED"**);**

**}**

**}**

module**.**exports **=** **{**

toDB**:** **function(**DataToDB**)**

**{**

writeToDB**(**DataToDB**,**"data"**);**

}

};

NetAppDemo.js :

**const** express **=** require**(**'express'**);**//express module

**const** request **=** require**(**'request'**);**

**var** data**={};**

**const** mongo **=** require**(**'./Mongo'**);**

**function** get\_hardware\_inventory**(**store\_hardware\_inventory**)**

**{**

request**(**'http://localhost:3001/storage-systems/5/hardware-inventory'**,** **{**json**:** **true},** **(**err**,** res**,** body**)** **=>** **{**

**if** **(**err**)** **{**

**return** console**.**log**(**err**+**"Im HereA"**);**

**}**

console**.**log**(**"got hardware-inventory"**);**

**return** store\_hardware\_inventory**(**body**);**

**});**

**}**

**function** get\_storage\_systems**(**store\_storage\_systems**)**

**{**

request**(**'http://localhost:3001/storage-systems'**,** **{**json**:** **true},** **(**err**,** res**,** body**)** **=>** **{**

**if** **(**err**)** **{**

**return** console**.**log**(**err**+**"Im HereB"**);**

**}**

console**.**log**(**"got storage-systems"**);**

**return** store\_storage\_systems**(**body**);**

**});**

**}**

**function** get\_failures**(**store\_failures**)**

**{**

request**(**'http://localhost:3001/storage-systems/5/failures'**,** **{**json**:** **true},** **(**err**,** res**,** body**)** **=>** **{**

**if** **(**err**)** **{**

**return** console**.**log**(**err**+**"Im HereC"**);**

**}**

console**.**log**(**"got failures"**);**

store\_failures**(**body**);**

**});**

**}**

**function** store\_hardware\_inventory**(**value**)**//callback function

**{**

data**[**0**]=**value**;**

get\_storage\_systems**(**store\_storage\_systems**);**

**}**

**function** store\_storage\_systems**(**value**)**//callback function

**{**

data**[**1**]=**value**;**

get\_failures**(**store\_failures**);**

**}**

**function** store\_failures**(**value**)**//callback function

**{**

data**[**2**]=**value**;**

print**();**

}

function print()

{

if(!data)

console.log("DATA EMPTY");

}

module.exports={

storeData: function()

{

var myPromise = new Promise(function(resolve,rejected){

get\_hardware\_inventory(store\_hardware\_inventory);

setTimeout(()=> resolve(data),2000);

});

myPromise.then((res)=>{

res=data;

mongo.toDB(res);

});

}

}

readFromMongo.js:

**const** express **=** require**(**'express'**);**//express module

**const** router **=** express**.**Router**();**//express-router module

**const** MongoClient **=** require**(**'mongodb'**).**MongoClient**;**//mongo module and client

**const** ObjectID **=** require**(**'mongodb'**).**ObjectID**;**//object ID-useful for data sorting

//connect block

**const** connection **=** **(**closure**)** **=>** **{**

**return** MongoClient**.**connect**(**'mongodb://localhost:27017/mydb'**,** **(**err**,** client**)** **=>** **{**

**if** **(**err**)** **return** console**.**log**(**err**);**

**let** db **=** client**.**db**(**'mydb'**);**

closure**(**db**);**

**});**

**};**

//error handling

**const** sendError **=** **(**err**,** res**)=>{**

response**.status** **=** 501**;**//status code

response**.**message **=** **typeof** err **==**'object' **?** err**.**message **:** err**;** //the error massage that include the error information

res**.status(**501**).**json**(**response**);**//return the status code and message

**};**

//response handling - the structure

**let** response**={**

**status:** 200**,**

data**:** **[],**

message**:** **null**

**};**

//get users

router**.**get**(**'/data'**,** **(**req**,** res**)** **=>** **{**

connection**((**db**)** **=>** **{**

db**.**collection**(**'data'**)**

**.**find**()**

**.**toArray**()**

**.**then**((**users**)** **=>** **{**

response**.**data **=** users**;**

res**.**json**(**response**);**

**})**

**.catch((**err**)** **=>** **{**

sendError**(**err**,** res**);**

**});**

**});**

**});**

module**.**exports **=** router**;**

Server.js:

**const** express **=** require**(**'express'**);**

**const** bodyParser **=** require**(**'body-parser'**);**

**const** path **=** require**(**'path'**);**

**const** http **=** require**(**'http'**);**

**const** app **=** express**();**

**const** NetAppData **=** require**(**'./server/routes/NetAppDemo'**);**

// API file for interacting with MongoDB

**const** MongoDBapi **=** require**(**'./server/routes/Mongo'**);**

**const** readFromMongo **=** require**(**'./server/routes/readFromMongo'**);**

**const** AuthUser **=** require**(**'./server/routes/AuthUser'**);**

**const** AdminPage **=** require**(**'./server/routes/AdminPage'**);**

**const** BugReport **=** require**(**'./server/routes/BugReport'**);**

**var** StillNeedUserInfo **=** **true;**

app**.**use**(function(**req**,** res**,** next**)** **{**

res**.**header**(**"Access-Control-Allow-Origin"**,** "\*"**);**

res**.**header**(**"Access-Control-Allow-Headers"**,** "Origin, X-Requested-With, Content-Type, Accept"**);**

next**();**

**});**

// Parsers

app**.**use**(**bodyParser**.**json**());**

app**.**use**(**bodyParser**.**urlencoded**({** extended**:** **false}));**

// Angular DIST output folder

app**.**use**(**express**.static(**path**.**join**(**\_\_dirname**,** 'dist'**)));**

app**.**use**(**'/readFromMongo'**,** readFromMongo**);**

app**.**use**(**'/AuthUser'**,** AuthUser**);**

app**.**use**(**'/AdminPage'**,**AdminPage**);**

app**.**use**(**'/BugReport'**,**BugReport**);**

// Send all other requests to the Angular app

app**.**get**(**'/snake'**,(**req**,**res**)=>{**

res**.**sendFile**(**path**.**join**(**\_\_dirname**,**'dist/assets/snake.html'**));**

**})**

app**.**get**(**'\*'**,** **(**req**,** res**)** **=>** **{**

res**.**sendFile**(**path**.**join**(**\_\_dirname**,** 'dist/index.html'**));**

**});**

//Set Port

**const** port **=** process**.**env**.**PORT **||** '3000'**;**

app**.**set**(**'port'**,** port**);**

**const** server **=** http**.**createServer**(**app**);**

server**.**listen**(**port**,** **()** **=>** **{**

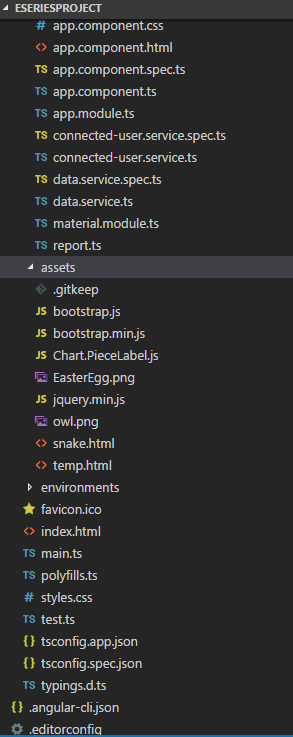
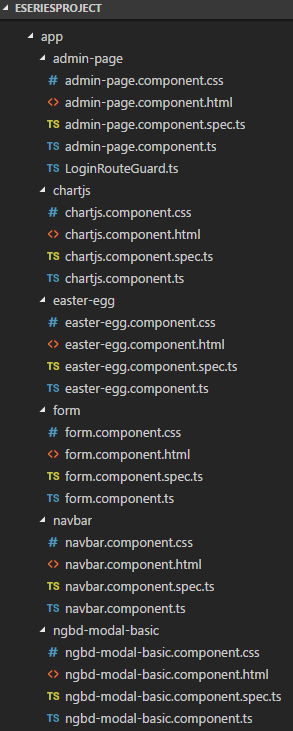
console**.**log**(**`Running on localhost:${port}`**);**

NetAppData**.**storeData**();**

**});**

# מבנה צד לקוח:

צד הלקוח מבוסס Angular 5 , שהיא מבוססת TypeScript וHTML. (TypeScript מבוסס JavaScript)זוהי הפלטפורמה המתקדמת ביותר כיום לפיתוח אפליקציות WEB. הפלטפורמה מורכבת מModules המוסיפים פונקציונליות ומComponents שהם בעצם רכיבים הבונים את ממשק המשתמש. כל Component מכיל קובץ .ts שהוא המחלקה, קובץ html שהוא המבנה, וקובץ css של עיצוב.



בפרוייקט שלי, צד הלקוח בנוי מפס ניווט (NavBar) קבוע, עליון ותחתון, וביניהם מסך המציג Components משתנים , Router-Outlet. בעצם לפי הניווט המערכת משנה את מה שמוצג על המסך מבלי לטעון מחדש את העמוד.

Admin-page: דף ניהול של המערכת, רק למשתמש שהוגדר אדמין יש גישה אליו, הקובץ LoginRouteGuard.ts מטפל באבטחת הדף.

Chartjs: Component המציג טבלאות פאי המייצגות כל אחת מערכת אחסון

Form: Component להצגת טופס

Navbar : Component המציג שורת ניווט קבועה

Ngbd-modal-basic: חלון קובץ גלובאלי של המערכת

App.component.ts: המחלקה הראשית

App.module.ts: קובץ המכיל imports למערכת

Connected-user.service.ts: שירות מערכת המנהל את המידע על המשתמש המחובר

Data.service.ts: שירות מערכת המנהל העברת מידע במערכת

Material.module.ts: imports למודולי עיצוב

Report.ts : מחלקה לאובייקט של דיווח

בתיקייה assets נמצאים קבצי עזר למערכת:

Bootstrap.js: עיצוב בוטסטראפ

Chart.pieceLabel.js: מאפשר כתיבה על פאי

EasterEgg+Owl: קבצי תמונה

Snake.html: משחק מוסתר

מחוץ לתיקייה זאת,

Index.html: הדף הראשוני של המערכת, שם נטענים כל הרכיבים ומשם מופעל מסך הloading

# Admin Page Component

Admin-page.component.ts:

**import** **{** Component**,** OnInit**,**AfterViewInit **}** **from** '@angular/core'**;**

**import** **{** DataService **}** **from** '../data.service'**;**

**import** **{** DialogService**}** **from** '../ngbd-modal-basic/ngbd-modal-basic.component'**;**

**import** **{** Input **}** **from** '@angular/core/src/metadata/directives'**;**

**import** **{** NavbarComponent **}** **from** '../navbar/navbar.component'

**import** **{** ConnectedUserService **}** **from** '../connected-user.service'

**import** **{** MatTabChangeEvent **}** **from** '@angular/material'**;**

@Component**({**

selector**:** 'app-admin-page'**,**

templateUrl**:** './admin-page.component.html'**,**

styleUrls**:** **[**'./admin-page.component.css'**]**

**})**

**export** **class** AdminPageComponent **implements** OnInit**,**AfterViewInit **{**

jsons**:** any**;**

showLoadingSpinner**:boolean=false;**

bugsFromDb**:**any**;**

adminlist**:**any**;**

userinfo**:** any**;**

userinfostring**:** string**;**

bugreports**:**string**;**

userName**:** any**;**

dataToshowA**:** string**;**

dataToshowB**:** string**;**

dataToshowC**:** string**;**

showDrivesBool**:** **boolean** **=** **false;**

showSystemsBool**:** **boolean** **=** **false;**

showFailuresBool**:** **boolean** **=** **false;**

showUserInfoBool**:** **boolean** **=** **false;**

showBugReportBool**:** **boolean** **=** **false;**

showBugsBool**:** **boolean** **=** **false;**

**constructor(** **private** \_dataService **:** DataService**,private** dialogService **:** DialogService**,** **private** userService**:** ConnectedUserService**)** **{**

**this.**\_dataService**.**getData**().**subscribe**(**data **=>** **{**

**this.**jsons **=** data

**this.**dataToshowA**=**data**.**data**[**0**][**0**];**

**this.**dataToshowB**=**data**.**data**[**0**][**1**];**

**this.**dataToshowC**=**data**.**data**[**0**][**2**];**

**});**

**this.**getUserInfo**()**

**this.**getbugreportstempfunction**()**

**}**

isAdminConnected**:boolean;**

ngOnInit**()** **{**

**}**

ngAfterViewInit**(){**

**}**

getbugreportstempfunction**(){**

**this.**\_dataService**.**getBugReports**().**subscribe**(**data **=>** **{**

this.bugsFromDb = data;

this.bugsFromDb = JSON.parse(this.bugsFromDb.\_body);

this.bugsFromDb = this.bugsFromDb.data;

})

}

getUserInfo()

{

var data = this.userService.getData();

this.isAdminConnected= data.A;

this.userName=data.C;

}

showDrives()

{

if(!this.showDrivesBool)

{

this.showSystemsBool=false;

this.showDrivesBool=true;

this.showFailuresBool=false;

this.showUserInfoBool=false;

this.showBugReportBool=false;

this.showBugsBool=false;

}

else

{

this.showDrivesBool=false;

}

}

showSystems()

{

if(!this.showSystemsBool)

{

this.showSystemsBool=true;

this.showDrivesBool=false;

this.showFailuresBool=false;

this.showUserInfoBool=false;

this.showBugReportBool=false;

this.showBugsBool=false;

}

else

{

this.showSystemsBool=false;

}

}

showFailures()

{

if(!this.showFailuresBool)

{

this.showSystemsBool=false;

this.showDrivesBool=false;

this.showFailuresBool=true;

this.showUserInfoBool=false;

this.showBugReportBool=false;

this.showBugsBool=false;

}

else

{

this.showFailuresBool=false;

}

}

showUserInfo()

{

if(!this.showUserInfoBool)

{

this.showSystemsBool=false;

this.showDrivesBool=false;

this.showFailuresBool=false;

this.showUserInfoBool=true;

this.showBugReportBool=false;

this.showBugsBool=false;

}

else

{

this.showUserInfoBool=false;

}

}

showBugReport()

{

this.dialogService.confirm("Enter Bug Details","");

if(!this.showBugReportBool)

{

this.showSystemsBool=false;

this.showDrivesBool=false;

this.showFailuresBool=false;

this.showUserInfoBool=false;

this.showBugReportBool=true;

this.showBugsBool=false;

}

else

{

this.showBugReportBool=false;

}

}

showBugs()

{

if(!this.showBugsBool)

{

this.showSystemsBool=false;

this.showDrivesBool=false;

this.showFailuresBool=false;

this.showUserInfoBool=false;

this.showBugReportBool=false;

this.showBugsBool=true;

}

else

{

this.showBugsBool=false;

}

}

doSomething(){

console.log("clicked")

}

}

Admin-page.component.html:

<mat-card>

<mat-card-content>

<h2 class=**"example-h2"**>**Hi Admin!**</h2>

<mat-tab-group class="demo-tab-group">

<mat-tab label="Home">

<div class=**"demo-tab-content"**>

**Admin Page**

</div>

</mat-tab>

<mat-tab label="Drives">

<div class=**"demo-tab-content"**>

<mat-card class="example-card">

<mat-card-header>

<mat-card-title>**Drives**</mat-card-title>

</mat-card-header>

<mat-card-content>

<ngx-json-viewer [json]="dataToshowA" [expanded]="false" [cleanOnChange]="true"></ngx-json-viewer>

</mat-card-content>

</mat-card>

</div>

</mat-tab>

<mat-tab label="Systems">

<div class=**"demo-tab-content"**>

<mat-card \*ngIf="dataToshowB" class="example-card">

<mat-card-header>

<mat-card-title>**Systems**</mat-card-title>

</mat-card-header>

<mat-card-content>

<ngx-json-viewer [json]="dataToshowB" [expanded]="false" [cleanOnChange]="true"></ngx-json-viewer>

</mat-card-content>

</mat-card>

</div>

</mat-tab>

<mat-tab label="Failures">

<div class=**"demo-tab-content"**>

<mat-card \*ngIf="dataToshowC" class="example-card">

<mat-card-header>

<mat-card-title>**Failures**</mat-card-title>

</mat-card-header>

<mat-card-content>

<ngx-json-viewer [json]="dataToshowC" [expanded]="false" [cleanOnChange]="true"></ngx-json-viewer>

</mat-card-content>

</mat-card>

</div>

</mat-tab>

<mat-tab label="Bugs">

**<div class="demo-tab-content">**

**<div \*ngFor="let Data of bugsFromDb; let i = index">**

**<mat-card class="example-card">**

**<mat-card-header>**

**<mat-card-title>Bug Report</mat-card-title>**

**<mat-card-subtitle>Name:{{bugsFromDb[i].name}}</mat-card-subtitle>**

**<mat-card-subtitle>Date:{{bugsFromDb[i].date}}</mat-card-subtitle>**

**</mat-card-header>**

**<mat-card-content>**

**<p>**

**{{bugsFromDb[i].description}}**

**</p>**

**</mat-card-content>**

**</mat-card>**

**</div>**

**</div>**

**</mat-tab>**

**</mat-tab-group>**

**</mat-card-content>**

**</mat-card>**

Admin-page.component.css:

**.**alert **{**

**width:300px;**

**}**

**.**example-card**{**

**margin-right: 10px;**

**margin-left: 10px;**

margin-block-start**: 10px;**

**}**

LoginRouteGuard.ts:

**import** **{** CanActivate **}** **from** '@angular/router'**;**

**import** **{** Injectable **}** **from** '@angular/core'**;**

**import** **{** ConnectedUserService **}** **from** '../connected-user.service'

@Injectable**()**

**export** **class** LoginRouteGuard **implements** CanActivate **{**

**constructor(private** userService **:** ConnectedUserService**)** **{}**

canActivate**()** **{**

console**.**log**(this.**userService**.**isAdminConnected**);**

**return** **this.**userService**.**isAdminConnected**;**

**}**

**}**

# Chart js Component

Chartjs.component.ts:

**import** **{** Component**,** OnInit**,**AfterViewInit **}** **from** '@angular/core'**;**

**import** **{** DataService **}** **from** '../data.service'**;**

**import** **{** Chart **}** **from** 'chart.js'**;**

**import** 'chart.piecelabel.js'**;**

**import** **{** ViewChildren **}** **from** '@angular/core'**;**

**import** **{** QueryList **}** **from** '@angular/core'**;**

**import** **{** error **}** **from** 'util'**;**

**import** **{** DialogService **}** **from** '../ngbd-modal-basic/ngbd-modal-basic.component'**;**

@Component**({**

selector**:** 'app-chartjs'**,**

templateUrl**:** './chartjs.component.html'**,**

styleUrls**:** **[**'./chartjs.component.css'**]**

**})**

**export** **class** ChartjsComponent **implements** OnInit**,** AfterViewInit **{**

@ViewChildren**(**"chartCanvas"**)** ChartCanvas**:** QueryList**<**any**>;**

jsons**:** any**;**//data of all systems

NumberOfSystems**:** Number**;**//variable to store number of connected systems

charts **=** **[];**//will save all of the charts data to render them

onlyChartData**=[];**//secondery array for the charts

i**=**0**;**

systemName**=[];**//array of systems names

**constructor(** **private** \_dataService **:** DataService**,** **private** dialogService **:** DialogService**)** **{**

**this.**\_dataService**.**getData**().**subscribe**(**data **=>** **{**//subscribing to the data service function

**this.**jsons **=** data

**this.**NumberOfSystems **=** **this.**jsons**.**data**[**0**][**1**].**systems**.**length//get number of systems

**this.**createChartsData**();**//call the function

**});**

**}**

ngOnInit**()** **{**

**}**

ngAfterViewInit**()**//function that called only after html render is complete

**{**

console**.**log**(this.**NumberOfSystems**);**

**this.**ChartCanvas**.**changes**.**subscribe**(**c **=>** **{**//subscribe to function that watch for changes of the canvas

**this.**ChartCanvas**.**toArray**().**forEach**((**item**)** **=>** **{**

console**.**log**(**item**+**"-----"**+this.**i**)**

**this.**createCharts**(this.**onlyChartData**,**item**,this.**i**,this.**dialogService**,this.**jsons**)**//build chart to each canvas

**})**

**this.**putDatainChart**()**//enter data

**});**

**}**

createChartsData**()**//initialize the charts array with basic chart data

**{**

console**.**log**(this.**NumberOfSystems**);**

**for(var** i **=**0**;** i**<this.**NumberOfSystems**;**i**++)**

**{**

**var** pie **={** // the configuration of chart

type**:** 'doughnut'**,**

data**:** **{**

labels**:** **[**"Disks"**,** "Mgmt"**,** "Hardware"**,** "FC"**,** "Vols&Pools"**],**

datasets**:** **[**

**{**

backgroundColor**:[**"#008000"**,**"#008000"**,**"#008000"**,**"#008000"**,**"#008000"**],**

data**:** **[**20**,**20**,**20**,**20**,**20**]**

**}**

**]**

**},**

options**:** **{**

pieceLabel**:** **{**

render**:** 'label'**,**

showZero**:** **true,**

fontSize**:** 12**,**

fontColor**:** '#fff'**,**

fontStyle**:** 'normal'**,**

fontFamily**:** "'Helvetica Neue', 'Helvetica', 'Arial', sans-serif"**,**

position**:** 'default'**,**

overlap**:** **true,**

**},**

**elements:** **{**

center**:** **{**

**text:** 'Desktop'**,**

color**:** '#36A2EB'**,** //Default black

fontStyle**:** 'Helvetica'**,** //Default Arial

sidePadding**:** 15 //Default 20 (as a percentage)

**}**

**},**

title**:** **{**

display**:** **true**

**},**

animations**:** **true,**

tooltips**:** **{**

enabled**:** **false**

**},**

legend**:** **{**

display**:** **false**

**}**

**}**

**};**

**this.**onlyChartData**.**push**(**pie**);**//store it

**}**

**}**

createCharts**(**pieData**,**ctx**,**index**,**dialogService**,**jsons**){**//create them and assign to canvas

console**.**log**(**ctx**.**nativeElement**.**children**[**0**].**children**[**1**].**children**[**0**]);**

**var** tempChart **=** **new** Chart**(**ctx**.**nativeElement**.**children**[**0**].**children**[**1**].**children**[**0**],**pieData**[**index**]);**

**document.**getElementById**(**ctx**.**nativeElement**.**children**[**0**].**children**[**1**].**children**[**0**].**id**).onclick** **=** **function(**evt**){**//get context of canvas

**var** activePoints **=** tempChart**.**getElementAtEvent**(**evt**);**

**var** firstPoint **=** activePoints**[**0**];**//set click listeners to charts and assign actions

**if(**firstPoint**!==** undefined**)**

**{**

**if** **(**firstPoint**.**\_index**==**0**)** **{**//if #0 pressed

console**.**log**(**"Drives: System #"**+**firstPoint**.**\_chart**.**config**.**options**.**title**.text);**

**if(**firstPoint**.**\_chart**.**config**.**data**.**datasets**[**0**].**backgroundColor**[**firstPoint**.**\_index**]==**"#ff0000"**)**//check if its red - yes, there is a fail

**{**

**var** NumOfFailDrives**;**

**var** errorID**;**

**var** failedDrive **=[];**

**for(var** x **=** 0**;**x**<**jsons**.**data**[**0**][**2**].**length**;**x**++)**

**{**

**if(**jsons**.**data**[**0**][**2**][**x**].**failureType**==**"diskPoolDriveFailure"**)**//check witch failure in order to show modal correctly

**{**

NumOfFailDrives **=** jsons**.**data**[**0**][**2**][**x**].**extraData**.**failedDrives**.**length**;**

console**.**log**(**NumOfFailDrives**);**

errorID**=**x**;**

**for(var** q **=** 0**;** q**<**NumOfFailDrives**;**q**++)**

**{**

failedDrive**.**push**({**tray**:**jsons**.**data**[**0**][**2**][**errorID**].**extraData**.**failedDrives**[**q**].**physicalLocation**.**locationPosition**,**slot**:**jsons**.**data**[**0**][**2**][**errorID**].**extraData**.**failedDrives**[**q**].**physicalLocation**.**slot**,status:**jsons**.**data**[**0**][**2**][**errorID**].**extraData**.**failedDrives**[**q**].status,**capacity**:**jsons**.**data**[**0**][**2**][**errorID**].**extraData**.**failedDrives**[**q**].**rawCapacity**,**interfaceType**:**jsons**.**data**[**0**][**2**][**errorID**].**extraData**.**failedDrives**[**q**].**interfaceType**.**driveType**,**id**:**jsons**.**data**[**0**][**2**][**errorID**].**extraData**.**failedDrives**[**q**].**productID**});**//the string to show

console**.**log**(**q**);**

**}**

**}**

**}**

dialogService**.confirm(**"Failed Drives"**,**JSON**.**stringify**(**failedDrive**));**//calling to function that open the modal

**}**

**}**

**if(**firstPoint**.**\_index**==**1**){**//if #1 pressed

console**.**log**(**"Mgmt: System #"**+**firstPoint**.**\_chart**.**config**.**options**.**title**.text);**

**if(**firstPoint**.**\_chart**.**config**.**data**.**datasets**[**0**].**backgroundColor**[**firstPoint**.**\_index**]==**"#ff0000"**)**

**{**

dialogService**.confirm(**"Mgmt"**,**""**);**//calling to function that open the modal

**}**

**}**

**if(**firstPoint**.**\_index**==**2**){**

console**.**log**(**"Hardware: System #"**+**firstPoint**.**\_chart**.**config**.**options**.**title**.text);**

**if(**firstPoint**.**\_chart**.**config**.**data**.**datasets**[**0**].**backgroundColor**[**firstPoint**.**\_index**]==**"#ff0000"**)**//if red

**{**

**var** outputMessage**=**""**;**

**for(var** x **=** 0**;**x**<**jsons**.**data**[**0**][**2**].**length**;**x**++)**//determine what failure

**{**

**if(**jsons**.**data**[**0**][**2**][**x**].**failureType**==**"drawerDegraded"**)**

**{**

outputMessage**+=**"drawerDegraded \n -----------------------"**;**

**}**

**if(**jsons**.**data**[**0**][**2**][**x**].**failureType**==**"usmUnreadableSectorsExist"**)**

**{**

outputMessage**+=**"\n usmUnreadableSectorsExist \n databasecount -> unreadablesectors \n -----------------------"**;**

**}**

**}**

dialogService**.confirm(**"Hardware"**,**outputMessage**);**//calling to function that open the modal

**}**

**}**

**if(**firstPoint**.**\_index**==**3**){**//is #3 pressed

console**.**log**(**"FC: System #"**+**firstPoint**.**\_chart**.**config**.**options**.**title**.text);**

**if(**firstPoint**.**\_chart**.**config**.**data**.**datasets**[**0**].**backgroundColor**[**firstPoint**.**\_index**]==**"#ff0000"**)**

**{**

dialogService**.confirm(**"FC"**,**""**);**

**}**

**}**

**if(**firstPoint**.**\_index**==**4**){**

console**.**log**(**"VolsPols: System #"**+**firstPoint**.**\_chart**.**config**.**options**.**title**.text);**

**if(**firstPoint**.**\_chart**.**config**.**data**.**datasets**[**0**].**backgroundColor**[**firstPoint**.**\_index**]==**"#ff0000"**)**

**{**

**var** outputMessage**=**""**;**

**for(var** x **=** 0**;**x**<**jsons**.**data**[**0**][**2**].**length**;**x**++)**//determine witch failure

**{**

**if(**jsons**.**data**[**0**][**2**][**x**].**failureType**==**"diskPoolReconstructionDriveCountBelowThreshold"**)**

**{**

outputMessage**+=**"diskPoolReconstructionDriveCountBelowThreshold -> Not Enough Spares \n Disk Pool:"**+**jsons**.**data**[**0**][**2**][**x**].**objectData**.**label**+**"\n Required Reserved Drive Count:"**+**jsons**.**data**[**0**][**2**][**x**].**objectData**.**volumeGroupData**.**diskPoolData**.**reconstructionReservedDriveCount**+**"\n Current Reserved Drive Count:"**+**jsons**.**data**[**0**][**2**][**x**].**objectData**.**volumeGroupData**.**diskPoolData**.**reconstructionReservedDriveCountCurrent**+**"\n -----------------------"**;**

**}**

**if(**jsons**.**data**[**0**][**2**][**x**].**failureType**==**"diskPoolCapacityDepleted"**)**

**{**

outputMessage**+=**"\n diskPoolCapacityDepleted \n Disk Pool:"**+**jsons**.**data**[**0**][**2**][**x**].**objectData**.**label**+**"\n Required Reserved Drive Count:"**+**jsons**.**data**[**0**][**2**][**x**].**objectData**.**volumeGroupData**.**diskPoolData**.**reconstructionReservedDriveCount**+**"\n Current Reserved Drive Count:"**+**jsons**.**data**[**0**][**2**][**x**].**objectData**.**volumeGroupData**.**diskPoolData**.**reconstructionReservedDriveCountCurrent**+**"\n Used Capacity:"**+**jsons**.**data**[**0**][**2**][**x**].**objectData**.**usedSpace**+**"\n Free Capacity:"**+**jsons**.**data**[**0**][**2**][**x**].**objectData**.**freeSpace**+**"\n -----------------------"**;**

**}**

**if(**jsons**.**data**[**0**][**2**][**x**].**failureType**==**"degradedVolume"**)**

**{**

outputMessage**+=**"\n degradedVolume \n Disk Pool:"**+**jsons**.**data**[**0**][**2**][**x**].**objectData**.**label**+**"Replace Disks Cyka Blyat \n -----------------------"**;**

**}**

**}**

dialogService**.confirm(**"Vols&Pools"**,**outputMessage**);**//calling to function that open the modal

**}**

**}**

**}**

**}**

**this.**charts**.**push**(**tempChart**);**//store data

**this.**i**++;**//store index

**}**

putDatainChart**()**//paint the charts if there is errors

**{**

**if(this.**jsons**)**

**{**

**for(var** k **=** 0**;** k**<this.**charts**.**length**;**k**++)**

**{**

**if(this.**jsons**.**data**[**0**][**1**].**systems**[**k**].status!=**"optimal"**)**//if there is failure in system the status will not be optimal

**{**

console**.**log**(this.**jsons**.**data**[**0**][**1**].**systems**[**k**].status);**

**for(var** x **=** 0**;**x**<this.**jsons**.**data**[**0**][**2**].**length**;**x**++)**//determine with slice paint to red

**{**

**if(this.**jsons**.**data**[**0**][**2**][**x**].**failureType**==**"diskPoolDriveFailure"**)**

**{**

console**.**log**(**k**+**"-diskToRed"**);**

**this.**charts**[**k**].**config**.**data**.**datasets**[**0**].**backgroundColor**[**0**]=**"#ff0000"**;**

**this.**charts**[**k**].**update**();**

**}**

**if(this.**jsons**.**data**[**0**][**2**][**x**].**failureType**==**"diskPoolReconstructionDriveCountBelowThreshold"**||this.**jsons**.**data**[**0**][**2**][**x**].**failureType**==**"diskPoolCapacityDepleted"**)**

**{**

console**.**log**(**k**+**"-volspolsToRed"**);**

**this.**charts**[**k**].**config**.**data**.**datasets**[**0**].**backgroundColor**[**4**]=**"#ff0000"**;**

**this.**charts**[**k**].**update**();**

**}**

**if(this.**jsons**.**data**[**0**][**2**][**x**].**failureType**==**"drawerDegraded"**||this.**jsons**.**data**[**0**][**2**][**x**].**failureType**==**"usmUnreadableSectorsExist"**)**

**{**

console**.**log**(**k**+**"-hardwareToRed"**);**

**this.**charts**[**k**].**config**.**data**.**datasets**[**0**].**backgroundColor**[**2**]=**"#ff0000"**;**

**this.**charts**[**k**].**update**();**

**}**

**}**

**}**

**this.**systemName**.**push**(this.**jsons**.**data**[**0**][**1**].**systems**[**k**].**name**);**//store system names

**}**

**}**

**else**

**{**

console**.**log**(**"jsons is empty"**);**

**}**

**}**

**}**

Chartjs.component.html:

<script src=**"../../assets/Chart.PieceLabel.js"**></script>

<div>

<table>

<tr>

<td #chartCanvas \*ngFor=**"let Data of onlyChartData; let i = index"**>

<mat-card class="example-card">

<mat-card-header>

<mat-card-title>**{{systemName[i]}}**</mat-card-title>

</mat-card-header>

<mat-card-content>

<canvas id=**"canvas{{i}}"** width=**"300"** height=**"300"**>**{{charts}}**</canvas>

</mat-card-content>

</mat-card>

</td>

</tr>

</table>

</div>

**Chartjs.component.css**

**.**example-card**{**

**margin-right: 10px;**

**margin-left: 10px;**

margin-block-start**: 10px;**

**}**

# Form Component

Form.component.ts:

**import** **{** Component **}** **from** '@angular/core'**;**

**import** **{** Report **}** **from** '../report'**;**

**import** **{** DataService **}** **from** '../data.service'**;**

@Component**({**

selector**:** 'app-form'**,**

templateUrl**:** './form.component.html'**,**

styleUrls**:** **[**'./form.component.css'**]**

**})**

**export** **class** FormComponent **{**

**constructor(** **private** \_dataService **:** DataService**)** **{**

**}**

model **=** **new** Report**(**'Arad'**,**'report'**,new** Date**);**

isSubmited**=false;**

submitted **=** **false;**

launchEasterEgg **=** **false;**

onSubmit**()** **{** //when submit button is clicked

**if(this.**model**.**description**!=**"AradTheKing"**)**

**{**

console**.**log**(this.**model**)**

**this.**submitted **=** **true;**

**}**

**else**

**{**

**this.**launchEasterEgg **=** **true;**

**}**

**}**

get diagnostic**()** **{**

**return** JSON**.**stringify**(this.**model**);**

**}**

submited**(){**//when submitted

**this.**isSubmited**=true;**

**this.**\_dataService**.**sendBugReport**(this.**model**).**subscribe**(**report**=>{**

console**.**log**(**"sent to server"**);**

**})**

**}**

**}**

Form.component.html:

<div class=**"container"**>

<form (ngSubmit)=**"onSubmit()"** #ReportForm=**"ngForm"**>

<div class=**"form-group"**>

<label for=**"name"**>**Name**</label>

<input type=**"text"** class=**"form-control"** id=**"name"** required [(ngModel)]=**"model.name"** name=**"name"**>

</div>

<div class=**"form-group"**>

<label for=**"Description"**>**Description**</label>

<input type=**"text"** class=**"form-control"** id=**"description"** required [(ngModel)]=**"model.description"** name=**"description"**>

</div>

<button type=**"submit"** class=**"btn btn-success"** (click)=**"submited()"**>**Submit**</button>

<a \*ngIf=**"isSubmited"**>**Sent**</a>

<a \*ngIf=**"launchEasterEgg"** href=**"/snake"** target=**"\_blank"**>

<img src=**"../../assets/EasterEgg.png"** width=**"50"** height=**"50"** alt=**"/snake"**/>

</a>

</form>

</div>

# NavBar Component

Navbar.component.ts:

**import** **{** Component**,** OnInit**,**AfterViewInit **}** **from** '@angular/core'**;**

**import** **{** DataService **}** **from** '../data.service'**;**

**import** **{** ConnectedUserService **}** **from** '../connected-user.service'

**import** **{** DialogService**}** **from** '../ngbd-modal-basic/ngbd-modal-basic.component'**;**

@Component**({**

selector**:** 'app-navbar'**,**

templateUrl**:** './navbar.component.html'**,**

styleUrls**:** **[**'./navbar.component.css'**]**

**})**

**export** **class** NavbarComponent **implements** OnInit**,**AfterViewInit **{**

userinfo**:** any**;**//store userinfo

userName**:** any**;**//store username

CurDate**:**any**;**//store data

adminlist**:**any**;**//store admin data from db

isAdminConnected**:boolean;**//is admin connected right now

**constructor(private** \_dataService **:** DataService**,private** userService **:** ConnectedUserService**,private** dialogService **:** DialogService**)** **{** **}**

ngOnInit**()** **{**//function that called when initialized

**this.**getDate**();**

**}**

ngAfterViewInit**(){**//function called when html render complete

**this.**getUserInfo**();**

**}**

getUserInfo**(){**//function that subscribe to user info function in the data service

**this.**\_dataService**.**getUserInfo**().**subscribe**(**data **=>{**

**this.**userinfo**=**JSON**.**parse**(**data**.**\_body**);**//parse it to JSON format

**this.**userName**=this.**userinfo**.**DomainName**+**"//"**+this.**userinfo**.**UserName**;**//build string to show

**this.**getadminlist**();**

**})**

**}**

getadminlist**(){**//function that subscribe to admin list function in the data service

**this.**\_dataService**.**getAdminList**().**subscribe**(**data **=>** **{**

**this.**adminlist **=** data**;**

**this.**adminlist**=**JSON**.**parse**(this.**adminlist**.**\_body**);**//parse to JSON format

**if(this.**adminlist**.**length**>**0**)**

**{**

**for(var** i**=**0**;**i**<this.**adminlist**.**length**;**i**++)**

**{**

**if(this.**adminlist**[**i**].**id**.**includes**(this.**userName**)){**//check if connected user defined as admin in db

console**.**log**(**"admin connected"**);**

**this.**isAdminConnected**=true;**

**this.**userService**.**setData**(this.**isAdminConnected**,false,this.**userName**);**//set the data to connected user service

**}**

**}**

**}**

**else**

**{**

console**.**log**(this.**userName**+**"NOT ADMIN, Compared to:"**+this.**adminlist**.**\_body**);**

**}**

**})**

**}**

getDate**()**//get current date

**{**

**this.**CurDate**=new** Date**().**toDateString**();**

**}**

BugReport**()**//send bug report with the function from dialogservice

**{**

**this.**dialogService**.confirm(**"Enter Bug Details"**,**""**);**

**}**

**}**

Navbar.component.html:

<nav class=**"navbar sticky-top navbar-dark bg-dark"**>

<a class=**"navbar-brand"** href=**"#"**>

<img src=**"../../assets/owl.png"** width=**"50"** height=**"50"** class=**"d-inline-block align-center"** alt=**""**/>**E-Series Dashboard**</a>

<ul class=**"navbar-nav mr-auto"**>

<li class=**"navbar-item"**>

<a class=**"nav-link"** routerLink=**"/main"** routerLinkActive=**"active"**>**Main**</a>

</li>

</ul>

<div ngbDropdown placement=**"bottom-right"** class=**"d-inline-block"**>

<a \*ngIf=**"userName; else elseBlock"** class=**"nav-link dropdown-toggle"** id=**"dropdownBasic1"** ngbDropdownToggle><font color=**"white"**>**{{userName}}**</font></a>

<ng-template #elseBlock><a class=**"nav-link dropdown-toggle"** id=**"dropdownBasic1"** ngbDropdownToggle><font color=**"white"**>**Guest**</font></a></ng-template>

<div ngbDropdownMenu aria-labelledby=**"dropdownBasic1"**>

<button \*ngIf=**"isAdminConnected"** routerLink=**"/admin"** class=**"dropdown-item"**>**Admin Page**</button>

<button class=**"dropdown-item"** (click)=**"BugReport()"**>**Bug Report**</button>

</div>

</div>

</nav>

<nav class=**"navbar fixed-bottom navbar-dark bg-dark"**>

<a class=**"navbar-brand mx-auto"**><font color=**"white"**>**{{CurDate}}**</font></a>

</nav>

Navbar.component.css:

**.**navbar**.**fixed-bottom**.**navbar-dark**.**bg-dark**{**

**height:50px;**

**}**

a**.**nav-link**.**active**{**

**color: #23a4c0 !important;**

**}**

# Ngbd-modal-basic Component:

Ngbd-modal-basic.component.ts:

//Import core

//Import ng-bootstrap dependencies

**import** **{** Injectable **}** **from** '@angular/core'**;**

**import** **{** Observable **}** **from** 'rxjs/Observable'**;**

**import** 'rxjs/add/observable/of'**;**

**import** 'rxjs/add/operator/do'**;**

**import** 'rxjs/add/operator/delay'**;**

**import** **{**NgbModal**,** NgbModalOptions**,** NgbActiveModal**,** ModalDismissReasons**}** **from** '@ng-bootstrap/ng-bootstrap'**;**

**import** **{**Component**,** Input**,** OnInit**,** ApplicationRef**,** ChangeDetectorRef**}** **from** '@angular/core'**;**

@Component**({**

template**:** `

<div class="modal-header">

<h4 class="modal-title">{{ title }}</h4>

<button type="button" class="close" aria-label="Close" (click)="activeModal.dismiss('Cross click')">

<span aria-hidden="true">&times;</span>

</button>

</div>

<div class="modal-body">

<div \*ngIf="isFailedDrives; else elseBlockA">

<table class="table">

<td>Tray</td><td>Slot</td><td>Status</td><td>Capacity</td><td>Interface Type</td><td>ID</td>

<tr \*ngFor="let Data of failedDrivesInfo; let i = index">

<td>{{failedDrivesInfo[i].tray}}</td>

<td>{{failedDrivesInfo[i].slot}}</td>

<td>{{failedDrivesInfo[i].status}}</td>

<td>{{failedDrivesInfo[i].capacity}}</td>

<td>{{failedDrivesInfo[i].interfaceType}}</td>

<td>{{failedDrivesInfo[i].id}}</td>

</tr>

</table>

</div>

<ng-template #elseBlockA><p \*ngIf="bugreportscreen; else elseBlockB"><app-form></app-form></ng-template>

<ng-template #elseBlockB><p \*ngIf="isVolsPools; else elseBlockC"><p \*ngFor="let data of message; let i = index">

{{message[i]}}

<br/>

</p></ng-template>

<ng-template #elseBlockC><p #inputTraget >{{message}}</p> </ng-template>

</div>

<div class="modal-footer">

<button mat-raised-button color="primary" (click)="activeModal.close(true)">Ok</button>

</div>

` //all of this code is actually the html file, but angular allow to write it here instead of a seperate file

**})**

**export** **class** NgbdModalBasicComponent **implements** OnInit**{**

@Input**()** title**;**

@Input**()** message**;**

isFailedDrives**:boolean=false;**

isVolsPools**:boolean=false;**

bugreportscreen**:boolean=false;**

failedDrivesInfo**:**JSON**;**

**constructor(public** activeModal**:** NgbActiveModal**,** **public** changeRef**:** ChangeDetectorRef**)** **{**

console**.**log**(**"DialogComponent construct"**);**

**}**

ngOnInit**()** **{**//initialize bools

console**.**log**(**"DialogComponent init"**);**

**if(this.**title**==**"Failed Drives"**)**

**{**

**this.**isFailedDrives**=true;**

console**.**log**(**"true"**);**

**}**

**if(this.**title**==**"Vols&Pools"**)**

**{**

**this.**isVolsPools**=true;**

console**.**log**(**"true"**);**

**}**

**if(this.**title**==**"Enter Bug Details"**)**

**{**

**this.**bugreportscreen**=true;**

console**.**log**(**"true"**);**

**}**

**}**

**}**

@Injectable**()**

**export** **class** DialogService **{**

**constructor(private** modalService**:** NgbModal**)** **{}**

**confirm(**callerTitle**,**callerMessage**)** **{**//the opening modal function

**const** modalRef **=** **this.**modalService**.open(**NgbdModalBasicComponent**,{**size**:** 'lg'**});**

**if(**callerTitle**==**"Failed Drives"**)**//determine header for modal

**{**

callerMessage**=**JSON**.**parse**(**callerMessage**);**

modalRef**.**componentInstance**.**failedDrivesInfo**=**callerMessage**;**

**}**

**if(**callerTitle**==**"Vols&Pools"**)**

**{**

modalRef**.**componentInstance**.**title **=** callerTitle**;**

modalRef**.**componentInstance**.**message **=** callerMessage**.**split**(**"\n"**);**

modalRef**.**componentInstance**.**changeRef**.**markForCheck**();**

**return** modalRef**.**result**;**

**}**

**else**

**{**

modalRef**.**componentInstance**.**title **=** callerTitle**;**

modalRef**.**componentInstance**.**message **=** callerMessage**;**

modalRef**.**componentInstance**.**changeRef**.**markForCheck**();**

**return** modalRef**.**result**;**

**}**

**}**

**}**

# App Component:

App.component.ts:

**import** **{** Component **}** **from** '@angular/core'**;**

**import** **{** DataService **}** **from** './data.service'**;**

**import** **{** DialogService **}** **from** '../app/ngbd-modal-basic/ngbd-modal-basic.component'

@Component**({**

selector**:** 'app-root'**,**

templateUrl**:** './app.component.html'**,**

styleUrls**:** **[**'./app.component.css'**]**

**})**

**export** **class** AppComponent **{**

**}**

App.component.html:

<html>

<body>

<div class=**"wrapper"**>

<app-navbar></app-navbar>

<div id=**"content"**>

<router-outlet></router-outlet>

</div>

</div>

</body>

</html>

App.module.ts:

**import** **{** BrowserModule **}** **from** '@angular/platform-browser'**;**

**import** **{** NgModule **}** **from** '@angular/core'**;**

**import** **{** AppComponent **}** **from** './app.component'**;**

**import** **{** HttpClientModule**,** HttpClient **}** **from** '@angular/common/http'**;**

// Import the Http Module and our Data Service

**import** **{** HttpModule **}** **from** '@angular/http'**;**

**import** **{** DataService **}** **from** './data.service'**;**

**import** **{** NavbarComponent **}** **from** './navbar/navbar.component'**;**

**import** **{** ChartjsComponent **}** **from** './chartjs/chartjs.component'**;**

**import** **{** BrowserAnimationsModule **}** **from** '@angular/platform-browser/animations'**;**

**import** **{** DialogService**,**NgbdModalBasicComponent **}** **from** './ngbd-modal-basic/ngbd-modal-basic.component'**;**

**import** **{**NgbModule**}** **from** '@ng-bootstrap/ng-bootstrap'**;**

**import** **{**FormsModule**}** **from** '@angular/forms'**;**

**import** **{** RouterModule**,** Routes **}** **from** '@angular/router'**;**

**import** **{** AdminPageComponent **}** **from** './admin-page/admin-page.component'**;**

**import** **{** FormComponent **}** **from** './form/form.component'**;**

**import** **{** LoginRouteGuard **}** **from** './admin-page/LoginRouteGuard'**;**

**import** **{** ConnectedUserService **}** **from** './connected-user.service'**;**

**import** **{** MaterialModule **}** **from** './material.module'

**import** **{** NgxJsonViewerModule **}** **from** 'ngx-json-viewer'**;**

**import** **{** FlexLayoutModule **}** **from** "@angular/flex-layout"**;**

**const** appRoutes**:** Routes **=** **[**

**{** path**:** 'main'**,** component**:** ChartjsComponent **},{** path**:** ''**,** component**:** ChartjsComponent **},**

**{** path**:** 'admin'**,** component**:** AdminPageComponent**,** canActivate**:[**LoginRouteGuard**]}**

**];**

@NgModule**({**

declarations**:** **[**

AppComponent**,**

NavbarComponent**,**

ChartjsComponent**,**

NgbdModalBasicComponent**,**

AdminPageComponent**,**

FormComponent**,**

**],**

imports**:** **[**

BrowserModule**,**

HttpModule**,**

HttpClientModule**,**

BrowserAnimationsModule**,**

NgbModule**.**forRoot**(),**

FormsModule**,**

MaterialModule**,**

RouterModule**.**forRoot**(**

appRoutes**,**

**{** enableTracing**:** **false** **}** // <-- debugging purposes only

**),**

FlexLayoutModule**,**

NgxJsonViewerModule

],

entryComponents:[

NgbdModalBasicComponent,

ChartjsComponent

],

providers: [DataService,

DialogService,

LoginRouteGuard,

ConnectedUserService

], // <-Add DataService

bootstrap: [AppComponent]

})

export class AppModule { }

material.module.ts:

**import** **{** NgModule **}** **from** '@angular/core'**;**

**import** **{**

MatButtonModule**,**

MatMenuModule**,**

MatToolbarModule**,**

MatIconModule**,**

MatCardModule**,**

MatTabsModule**,**

MatProgressSpinnerModule

**}** **from** '@angular/material'**;**

@NgModule**({**

imports**:** **[**

MatButtonModule**,**

MatMenuModule**,**

MatToolbarModule**,**

MatIconModule**,**

MatCardModule**,**

MatTabsModule**,**

MatProgressSpinnerModule

**],**

exports**:** **[**

MatButtonModule**,**

MatMenuModule**,**

MatToolbarModule**,**

MatIconModule**,**

MatCardModule**,**

MatTabsModule**,**

MatProgressSpinnerModule

**]**

**})**

**export** **class** MaterialModule **{}**

report.ts:

**export** **class** Report **{**

**constructor(**

**public** name**:** string**,**

**public** description**:** string**,**

**public** date**:** Date

**)** **{** **}**

**}**

# Services:

Data.service.ts:

**import** **{** Injectable **}** **from** '@angular/core'**;**

**import** **{** Observable **}** **from** 'rxjs/Observable'**;**

**import** **{** catchError**,** retry **}** **from** 'rxjs/operators'**;**

**import** **{** HttpErrorResponse **}** **from** '@angular/common/http'**;**

**import** **{** Http **}** **from** '@angular/http'**;**

**import** **{** HttpClient **}** **from** '@angular/common/http'

**import** **{** HttpHeaders **}** **from** '@angular/common/http'**;**

**import** 'rxjs/add/operator/map'**;**

**import** **{** Response **}** **from** '@angular/http/src/static\_response'**;**

**import** **{** ErrorObservable **}** **from** 'rxjs/observable/ErrorObservable'**;**

**import** **{** Report **}** **from** './report'**;**

**const** httpOptions **=** **{**

headers**:** **new** HttpHeaders**({**

'Content-Type'**:** 'application/json'**,**

'Authorization'**:** 'my-auth-token'

**})**

**};**

@Injectable**()**

**export** **class** DataService **{**

resultA**:**any**;**

tempString**:**any**;**

**constructor(private** \_http**:** Http**,private** httpclient**:** HttpClient**)** **{** **}**

**private** handleError**(**error**:** HttpErrorResponse**)** **{**

**if** **(**error**.**error **instanceof** ErrorEvent**)** **{**

// A client-side or network error occurred. Handle it accordingly.

console**.**error**(**'An error occurred:'**,** error**.**error**.**message**);**

**}** **else** **{**

// The backend returned an unsuccessful response code.

// The response body may contain clues as to what went wrong,

console**.**error**(**

`Backend returned code ${error.status}, ` **+**

`body was: ${error.error}`**);**

**}**

// return an ErrorObservable with a user-facing error message

**return** **new** ErrorObservable**(**

'Something bad happened; please try again later.'**);**

**};**

getData**()** **{**//send http get to server to get data, map it and parse it to json format

console**.**log**(**"asking for data"**);**

**return** **this.**\_http**.**get**(**"/readFromMongo/data"**).**map**(**res **=>** res**.**json**())**

**}**

getUserInfo**(){**//send http get to server

console**.**log**(**"asking for user info"**);**

**this.**tempString **=** **this.**\_http**.**get**(**"/AuthUser/UserInfo"**)**

**return** **this.**tempString**;**

**}**

sendBugReport**(**report**:**Report**):** Observable**<**Report**>{**//send http post to server

console**.**log**(**"sending bug report to server"**);**

**return** **this.**httpclient**.**post**<**Report**>(**"/BugReport/SaveReport"**,** report**,** httpOptions**)**

**.**pipe**(**

catchError**(this.**handleError**)**

**);**

**}**

getBugReports**(){**//send http get to server

console**.**log**(**"asking for bug reports"**);**

**return** **this.**\_http**.**get**(**"/BugReport/AllBugs"**);**

**}**

getAdminList**(){**//send http get to server

console**.**log**(**"asking for admin list"**);**

**return** **this.**\_http**.**get**(**"/AuthUser/UserInfo/Admins"**);**

**}**

**}**

Connected-user.service.ts:

**import** **{** Injectable **}** **from** '@angular/core'**;**

@Injectable**()**

**export** **class** ConnectedUserService **{**

**constructor()** **{** **}**

isAdminConnected**:boolean;**

isGuest**:boolean;**

UserName**:**any**;**

setData**(**isAdminConnected**,**isGuest**,**UserName**)**

**{**

**this.**isAdminConnected**=**isAdminConnected**;**

**this.**isGuest **=** isGuest**;**

**this.**UserName **=** UserName**;**

**}**

getData**(){**

**var** array**={**A**:null,**B**:null,**C**:null};**

array**.**A**=this.**isAdminConnected**;**

array**.**B**=this.**isGuest**;**

array**.**C**=this.**UserName**;**

**return** array**;**

**}**

**}**