המכללה האקדמית אשקלון

המחלקה למדעי המחשב

פרויקט גמר בנושא:

נגישות במודל – המחלקה להוראת המדעים





מגישים:

דמיטרי לזינסקי 321050924

אורן שמואל 206225476

תמיר חג'ג' 312170038

מנחה פרויקט: פרופ' אסתר דוד

מנחה בתעשייה: נדב קבלרצ'יק

תוכן עניינים

1. מסמך דרישות:

1.1 הצגת הבעיה - הנגשת מוגבלות, מהו אתר נגיש

1.2 מטרת הפרויקט

1.3 דרישות פונקציונליות

2. מסמך עיצוב:

2.1 פתרונות אפשריים והנמקה לפתרון הנבחר

3. מהות הפרויקט:

3.1 שלבים לפתרון הבעיה

3.1.1 היכרות עם Axe – תיאורטית

3.1.2 היכרות עם Axe – מעשית (תוסף, בדיקות צד שרת, POC)

4. הכנות:

4.1 ההכנות שבוצעו בטרם יצירת והטמעת הפלאגין למערכת

4.1.1 הקמת Moodle מקומי

4.1.2 הקמת Moodle AWS

תוכן עניינים

5. תהליך כתיבת הפלאגין והטמעתו במודל:

5.1 הקמת פלאגין ריק

5.2 הטמעת סיפריית Axe במודל

5.3 הפעלת פופ-אפ בJQuery – צד לקוח

5.3.1 מטודות של JQuery

5.4 כתיבה לDB והצגתו - צד שרת

5.4.1 הקמת DB (install)

5.4.2 עדכון הDB

5.4.3 AJAX בצד לקוח

6. ספריות ומבנה הקוד:

6.1 GRUNT

6.2 YUI

6.3 AMD

6.4 PHP & JS

6.5 JQuery

6.6 AJAX

6.7 Axe

6.8 Maria DB

מסמך דרישות

הצגת הבעיה:

הנגשת מוגבלות

לאנשים עם מוגבלות זכות להשתתפות מלאה ושוויונית בכל תחומי החיים. חוק שוויון זכויות לאנשים עם מוגבלות ותקנות הנגישות נועדו להבטיח, שיוכלו לממש זכות זו.

תקנות הנגישות לשירות עוסקות בהסרת חסמים כדי שאנשים עם מוגבלות יוכלו לקבל שירות באותה רמת נוחות ואיכות, כפי שמקבל כלל הציבור, ויוכלו לנהל חיים עצמאיים ומלאים ובלי תלות בזולת. תקנות הנגישות לשירות עוסקות במגוון רחב של תחומים וביניהם: הנגשה פיזית של המקומות בהם ניתן השירות, התאמה של הדרכים והאמצעים בהם ניתן השירות, האופן שבו נמסר מידע לציבור וכן הנגשה של ערוצי השירות השונים.

ספקי שירות רבים עושים שימוש באינטרנט כערוץ למתן שירות לציבור (לדוגמא, לצורך מכירת מוצרים), וכן לשם מתן מידע אודות השירותים אותם הם מספקים. כדי שציבור האנשים עם מוגבלות יוכל לקבל שירות באמצעות האינטרנט כשאר הציבור, **יש לבצע התאמות באתרים ובאפליקציות המספקים את השירות או המידע אודותיו** לפי סימן ג' לתקנות שוויון זכויות לאנשים עם מוגבלות (התאמות נגישות לשירות), התשע"ג-2013

* הנגשת אתרי אינטרנט מתוך [אתר משרד המשפטים](https://www.gov.il/he/departments/guides/website_accessibility)

מהו אתר נגיש

אתר מונגש לפי תקנות הנגישות לשירות הוא אתר שבו בוצעו התאמות לפי תקן ישראלי 5568.

התקן הישראלי הינו אימוץ של הנחיות בינלאומיות - הנחיות נגישות לתכני אינטרנטWeb Content Accessibility Guidelines של ארגון ה W3C הבינלאומי, מסמך ההנחיות כולל קישורים להסברים ודוגמאות כיצד לבצע את ההנגשה.

התאמות הנגישות מאפשרות לאנשים עם מוגבלויות שונות כגון עיוורים ומוגבלי ראיה, אנשים עם מוגבלות פיזית המתקשים להפעיל את ידיהם, אנשים עם מוגבלויות קוגניטיביות כגון דיסלקסיה והפרעות קשב וריכוז, אנשים עם מוגבלות שמיעה ואנשים עם מוגבלויות אחרות, לעשות שימוש אפקטיבי בשירותים ובמידע המוצעים באתר.

לפניכם מספר דוגמאות להנחיות על פיהן מותאמים אתרים לאנשים עם מוגבלות:

* משתמש עיוור עושה שימוש בתוכנה שיש ביכולתה להקריא מידע מהמחשב. כאשר הוא גולש באתר הוא יוכל לקרוא את כל הטקסטים בסדר הגיוני (קריטריון 1.3.2), לקבל תיאור של מידע המצוי בתמונות (קריטריון 1.1.1), לזהות כותרות וקישורים (קריטריון 1.3.1) ולהיות מסוגל להפעיל את כל מה שיש באתר כגון קישורים, טפסים, כפתורים (קריטריונים 2.1.1, 2.1.2). **התאמות אלו דורשות ברובן כתיבת קוד תקני לאתר ומתן מידע חלופי לתמונות ולמולטימדיה**.
* גולשת שעקב מוגבלותה אינה יכולה להפעיל עכבר או מתקן דומה, תוכל לנווט באתר ולבצע פעולות באמצעות פקודות הניתנות דרך המקלדת (קריטריונים 2.1.1, 2.1.2, 2.4.3, 2.4.7).
* הטקסט באתר הוא בצבע בולט יחסית לרקע, כך שמתאפשר לאנשים שראייתם לקויה לקרוא את הכתוב (קריטריון 1.4.6).
* אנשים עם לקויות שמיעה יכולים לקבל את המידע המילולי הנמסר בסרטי וידאו שבאתר, על ידי הוספת כתוביות לסרטונים.
* הנגשת אתרי אינטרנט מתוך [מהי נגישות באינטרנט?](https://www.gov.il/he/departments/guides/website_accessibility?chapterIndex=2)

מטרת הפרויקט:

במסגרת קורס תכנות יישומי, פרויקט הסיום הוא ביישום פתרונות נגישות למערכת PeTeL

(Personalized Teaching and Learning Environment)

מערכת פטל מבוססת על פלטפורמת מודל קוד פתוח. זוהי מערכת לניהול למידה LMS

(Learning Management System)

לשם פיתוח רכיב זה יש צורך להכיר את מערכת Moodle ע"י

* Moodle Plugin Development – זהו קורס מנחה לכתיבת רכיבים במודל.
* Moodle APIs and Moodle Docs – הדוקומנטציה של מודל דב.

במסגרת הפרויקט אנחנו רוצים ליישם טכנולוגיות לפתרונות נגישות כדי לייעל ולעזור למורים המשתמשים במערכת בהנגשת הפעילויות והתכנים.

מטרת הפרויקט היא ליצור כלים המאפשרים למורים ליצור חומרים ופעילויות שעומדים בתקני הנגישות.

יעדים: ניישם כלים לבדיקת נגישות.

לשם כך כתבנו רכיב שעושה סקירה על התוכן המוזן של מורים, מציין את שגיאות הנגישות הקיימות ומראה להם היכן הם נמצאים ומחזיק בטבלה עם מידע סטטיסטי על השגיאות שבוצעו על ידי המורים.

למעשה קיים רכיב במודל אשר עושה ניטור על שגיאות נגישות בהזנת התוכן -אולם הוא אינו מכיר אלא רק שישה חוקי נגישות בלבד כשלמעשה קיימים כבר מעבר ל200 חוקים ואין אפשרות לעדכן את החוקים הקיימים במערכת המיושנת שיש כיום.

יש צורך במערכת שמתעדכנת אוטומטית בהתאם לחוקי הנגישות שמתעדכנים באופן שוטף. בנוסף יש צורך להגדיר רמת נגישות (AA לעומת AAA שיותר מחמיר וכן הלאה) בצורה מהירה ופשוטה בהתאם לקהל אליו המורה כותב את התוכן.

דרישות פונקציונליות

המטרה שלנו כאשר מורה יזין תוכן בכתבן (Atto Editor) יהיה לו כפתור שדרכו יוכל לנטר כמה שגיאות הוא עשה, היכן הם נמצאות ואיך אפשר לתקן אותן.

לשם כך יש צורך להכין פלאגין נגישות שידע על חוקי הנגישות ברשת , יעשה ניטור על התוכן המוזן ויראה איזו שגיאות קיימות בתוכן.

המטרה שהרכיב לא יהיה אוטומטי אלא רק בדרישה של המורה, תהיה אפשרות לעשות בדיקה על התוכן ולייצא לו פלט שגיאות.

במקביל לכך נדאג לייצור דוח ניטור (על ידי צד שרת) לאדמין המערכת.

לשם כך יש צורך להשתמש במנוע דינאמי אשר יודע לעשות את זה, קיימים כמה ברשת (LightHouse, axe, wave) אולם העדפה שזה יהיה מובנה למערכת Moodle ללא תלות בCDN (רשת למסירת תוכן (CDN) מתייחסת לקבוצת שרתים המפוזרת גיאוגרפית אשר פועלים יחד כדי לספק אספקה ​​מהירה של תוכן אינטרנט.)

יש לשם כך לייצר בתוך הפלאגין ספריה אשר יהיה אפשר בקלות לעדכן אותה כל פעם באופן ידני בשעת הצורך.

מסמך עיצוב

פתרונות אפשריים והנמקה לפתרון הנבחר

הוצע לנו מספר מנועים לשימוש בפלאגין שהיה עלינו לעשות, אולם החלטנו לעבוד עם המנוע Axe-Core בהמלצת המנחה נדב קבלרצ'יק.

היתרונות של המנוע הנ"ל:

* קיים הרבה שנים בתעשייה.
* יש לו קהילת מפתחים גדולה ועשירה.
* מתעדכן באופן שוטף.
* מכיל מעל ל200 חוקי נגישות.
* תואם לסטנדרטיים הבינלאומיים (Double A, Triple A).

הומלץ לנו להשתמש בJQuery לשימוש בפופ-אפ לצורך הפלט למשתמש

ובעזרת AJAX לייצר דוחות בצד שרת.

בנוסף התבקשנו להשתמש בפלאגין Configurable Report של מודל לצורך נראות טובה למסד הנתונים והצגת הדוח לאדמין המערכת.

מהות הפרויקט

המטרה להקים מערכת מודל בשרת AWS - Amazon EC2לייצר לשמה את הפלאגין ולקיים סביבה של מודל תחת שרת Production אמיתי ולייצר סביבה נכונה לפיתוח.

לשם כך הקמנו חשבון באמזון והקמנו שרת של AWS, התקנו על השרת את המודל ע"י שירות BITNAMI – זהו שירות אשר נותן לנו סביבות עבודה של קוד פתוח (כגון Red Mine,WordPress,Maria DB וכמובן Moodle)

לאחר שייצרנו את הפלאגין והשתלנו אותו במקום בתוך הAtto Editor של מודל עשינו Deploy לשרת.

**שלבים לפתרון הבעיה:**

היכרות עם Axe – תיאורטית

Axe הוא מנוע לבדיקת נגישות לאתרים וממשקי משתמש אחרים המבוססים על HTML. הוא פותח ע"י deque ע"י מעל 150 מפתחים, הוא בשימוש ע"י מעל 450 חברות ונעשה לו שיתוף ותרומות בGit באופן יומיומי.

בנוסף לכך הוא מהיר, מאובטח, קל משקל, והוא נבנה כדי להשתלב בצורה חלקה עם כל סביבת בדיקות קיימת, כך שתוכל להפוך את בדיקות הנגישות לאוטומטיות.

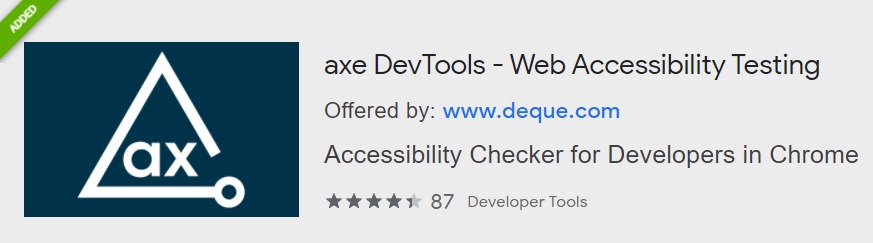
כללי הנגישות של Axe

ל- Ax-core סוגים שונים של כללים, עבור WCAG 2.0 ו- 2.1 ברמה A ו- AA, כמו גם מספר שיטות עבודה מומלצות המסייעות לזהות שיטות נגישות נפוצות.

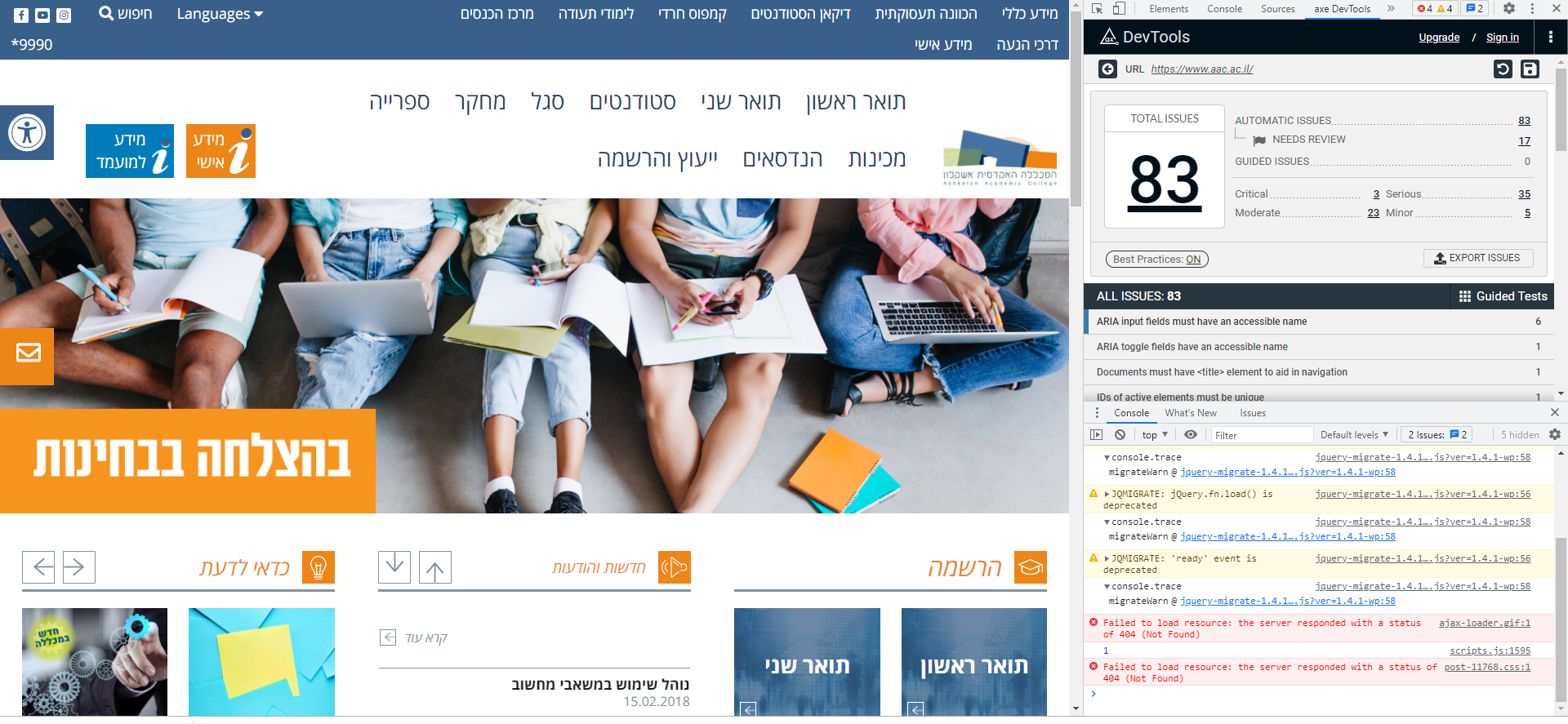
עם Axe-Core אפשר למצוא בממוצע 57% מבעיות WCAG באופן אוטומטי.

היכרות עם Axe – מעשית

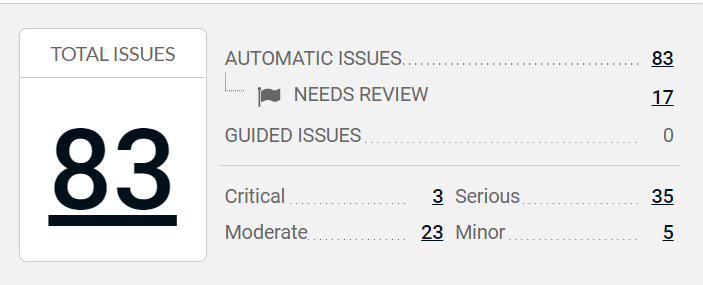
* התקנת התוסף לדפדפן Axe:



השתמשנו בתוסף של ה Axe על אתרי אינטרנט שונים עם הגדרות חוקי נגישות בסטנדרטים שונים כדי לחקור איך המנוע עובד.

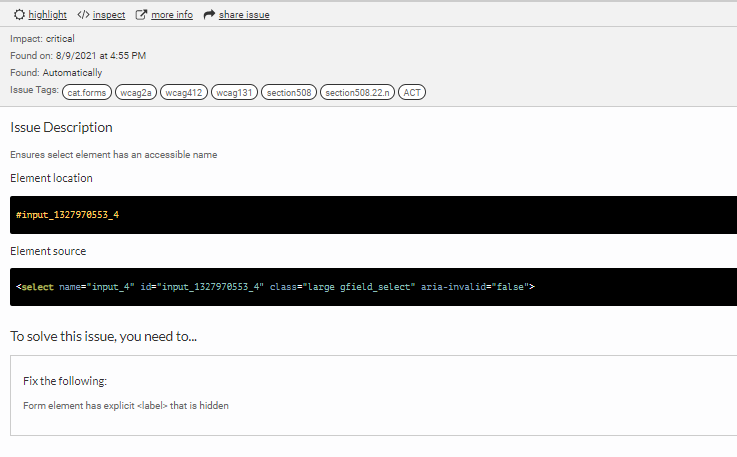


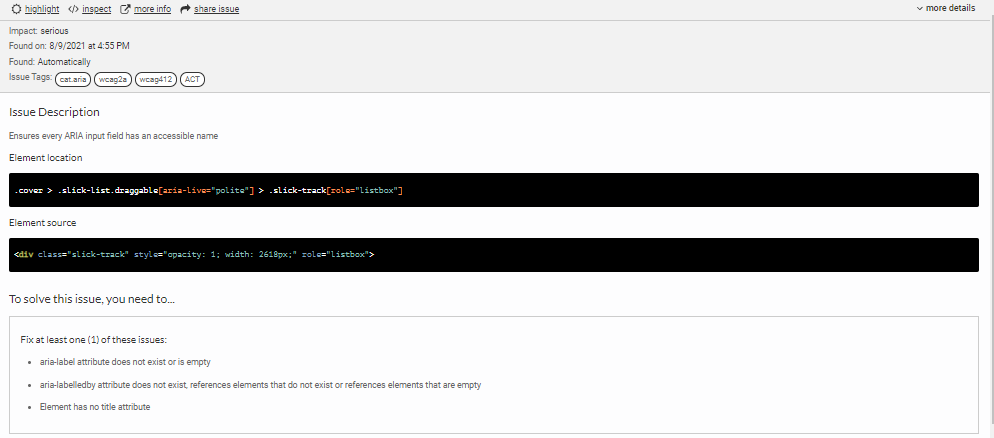
למעשה ישנן 83 בעיות של נגישות באתר המכללה:



כאשר נכנס לכל בעיה הוא יציג באופן מפורט מה לא תקין ואיך אפשר לתקן זאת

לדוגמא:





* מימוש Axe בעזרת Selenium WebDriver:

Selenium WebDriver היא פלטפורמה חינמית (open source) לבדיקות אוטמציה.

Selenium WebDriver הוכנסה כסטנדרט לתקינה של W3C.

ה-WebDriver הוא interface שהיום כל מפתח של דפדפן מממש אותו ולכן יש לנו API לעבוד מול כל הדפדפנים הקיימים. אנחנו כותבים פעם אחת ומריצים את הבדיקות מול כל הדפדנים ללא צורך לכתוב קוד יחודי לכל דפדפן.

ה-Selenium WebDriver חושף לנו 2 Interfaces עיקריים:

1. WebDriver – מייצג את העבודה מול הדפדפן (פתיחת דפדפן, גלישה לאתרים, back, forward וכו’).
2. WebElement – מייצג את העבודה מול האלמנטים על הדף. עבודה אל מול ה- (DOM (Document Object Module.

יצרנו דף HTML פשוט עם מספר בעיות נגישות ידועות, כתבנו קוד בשימוש עם הAxe ע"י פלטפורמת Selenium WebDriver-js אשר מנתחת את התוכן של האתר ומציגה בקובץ JSON את האובייקט החוזר ע"י הAxe המפרטת את בעיות הנגישות באותו עמוד HTML.

הפלט אשר נוצר לנו מהבדיקה:



שקלנו להשתמש Seleniumכגורם מקשר בין הפלאגין למנוע הAxe, אך לאחר ייעוץ עם המנחה שלנו הוחלט לוותר על הרעיון ולעבוד ישירות עם ספריית מנוע Axe ללא גורמים מקשרים.

* מימוש Axe ישירות דרך ספריית Axe על

POC (Proof of Concept) בעזרת תיבת דיאלוג JQuery:

בהמשך להמלצתו של המנחה, יצרנו אתר Demo עם בעיות נגישות ידועות. יצרנו כפתור המפעיל את מנוע הAxe המנתח את העמוד ומציג תיבת דיאלוג JQuery שבה מפורטות כל הבעיות הנגישות שישנן בעמוד.

על מנת לעשות זאת היינו צריכים לחקור וללמוד JQuery.

הראנו את התוצאה למנחה, והוא אישר שהפעם הDemo הנ"ל מקובל עליו לשימוש במערכת Moodle, אך אמר שהדרך לביצוע הimport לספריית הAxe אינה מתקבלת מכיוון שמערכת Moodle צריכה לעבוד גם במצב Offline ללא גישה אינטרנטית לCDN.

דבר זה חשוב לציון מכיוון שהטמעת ספרייה חיצונית חדשה למאגר הספריות הסגור בתוך הMoodle אינה פעולה סטנדרטית ודורשת עבודה רבה ותמיכה של אנשי המודל העולמי (Andrew Lyons – אדמין מערכת מודל העולמי) עזרו לנו בכך.

הכנות

ההכנות שבוצעו בטרם יצירת והטמעת הפלאגין למערכת:

* הקמת Moodle מקומי:

הכרנו את תוכנת Wampp :

תוכנה אשר מדמה סביבת שרת של PHP לוקאלי עם הקמת מסד נתונים Maria DB אשר תספק לנו את הפלטפורמה למודל המקומי.

היה צורך לקנפג את קובץ php.ini אשר יתאים לנו לסביבה שאנו רוצים (כמות זיכרון, מסד נתונים של sqli , נפח זיכרון, ואיזה extensions נקבל מהשרת)

לאחר מכן התקנו את המודל המקומי בגרסה 3.9 שזוהי הגרסה הכי חדשה ויציבה שקיימת למודל כיום , אשר עובדת היום בProduction במכון ויצמן – שנוכל לעבוד באופן מתואם איתם.

הקמנו קורסי בדיקה ופעילויות לראות שהסביבה תקינה.





* הקמת Moodle AWS:

בעקבות הוראתו של המנחה שכרנו שרת של AWS מאמזון תחת שירות של EC2 שיאחסן את המודל הגלובאלי שלנו, ונגבה אליו את ההתקדמות שלנו בשרתים הלוקאלים שיצרנו כל אחד במחשב שלו.

לשם כך השתמשנו בשירות הנקרא BITNAMI אשר דיברנו עליו לעיל.

הגיבוי לשרת נעשה בסיום כל יום עבודה שלנו על הפרויקט במכון ויצמן למדע.





תהליך כתיבת הפלאגין והטמעתו במודל

הקמת פלאגין ריק

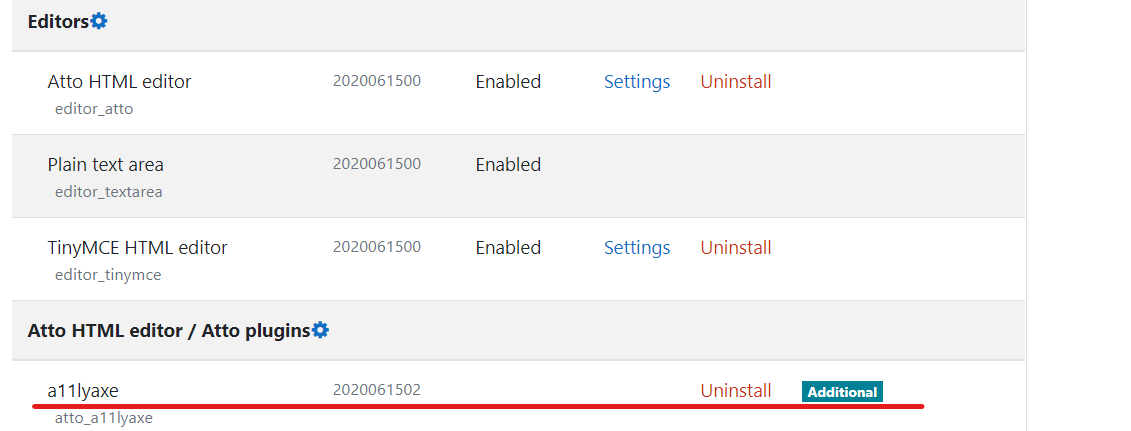
לצורך הקמת הפלאגין הריק היה צורך להכיר את הכתבן שאיתו עובדים במודל.

לכתבן יש כמה פלאגינים קיימים, חקרנו אותם (נמצאים בתיקיית : wamp64\www\moodle\lib\editor\atto\plugins)

והכנסו את הפלאגין שלנו לתוך התיקייה. יצרנו שני קבצים בסיסים אשר יש צורך בהם לכל פלאגין

1. Version – הגרסא שבה הפלאגין קיים, השם שלו להתקנה ומה הדרישות למודל (אם יש גרסה יותר ישנה אז המערכת תתריע על כך)
2. תיקיית Lang שבה קיימים הקבצים של השפה, במקרה שלנו לא התבקשנו לייצר בנק מחרוזות לכן קיים אך ורק בשפה האנגלית בלבד.

לאחר מכן הרמנו את ההתקנה וראינו שקיים פלאגין חדש למערכת בצורה תקינה (atto\_a11lyaxe) המילה Atto שמורה לכתבן שנדע שזה פלאגין ששייך לאוסף הפלאגינים של הכתבן והמילה a11lyaxe מסמנת את המילה נגישות בשפה המקצועית ואת המנוע שאיתו עבדנו – Axe.



הטמעת ספריית Axe במודל

היה לנו בעיה להטמיע את הספרייה של הAxe מאחר שאין אפשרות להטמיע CDN בתוך המודל כמו שהיה בPOC.

בפעם הראשונה המנחה לימד אותנו להשתמש בדרך תפירה מסוימת אולם לאחר עבודה רבה היא לא צלחה,

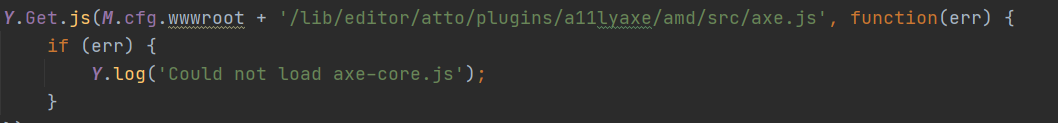
בדרך הנ"ל יש שלושה קבצים 1.קובץ הספרייה 2. תפירה לתוך הYUI 3. כניסה לתוך הקובץ של הכפתור.

זה לא צלח מאחר ולא הייתה אפשרות לקרוא לAxe עם המטודות שלו, כלומר לא יכולנו להביא את המטודה run שהיא מריצה את בדיקת הנגישות.

נאלצנו לתקשר עם השירות של מודל העולמי – בעזרת המנחה, שקישר אותנו בטלגרם (קהילת Moodle Dev) על מנת להוסיף את Axe לאוסף הספריות הקיימות במודל.

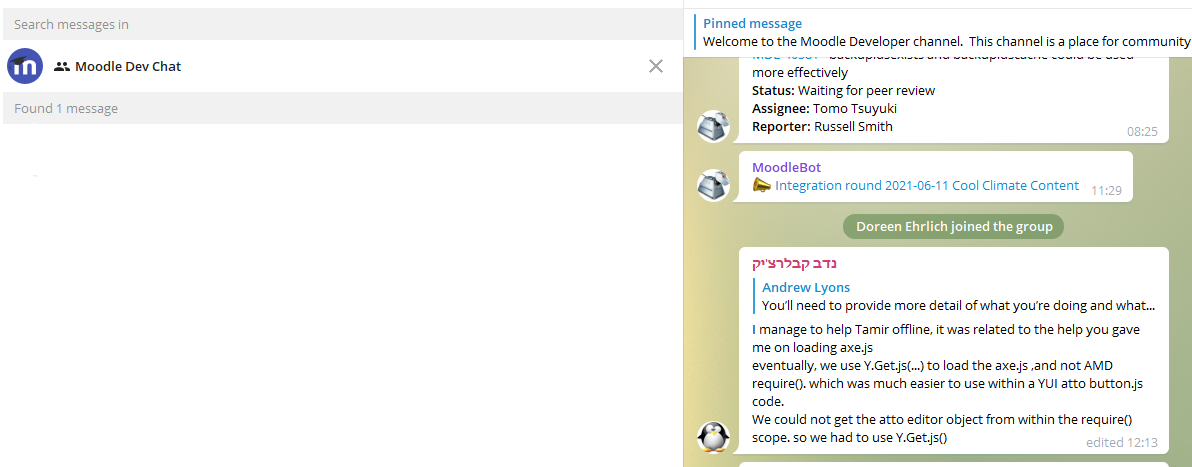
קיבלנו מהם שתי עצות:

1. בקשה שהמנוע יהיה קיים בתוך הפלאגין עצמו שיוכלו להשתמש בו במקום אחד ולהטמיע אותו רק מהתקנה פשוטה, ללא צורך בקצים נוספים מלבד התיקייה של הפלאגין.
2. קיבלנו את התפירה (קישוריות) לתוך תיקיית הפלאגין בדרך הנ"ל:



למעשה אנו קוראים לFrameWork הנקרא YUI אשר יש בו שימוש נרחב במערכת המודל ונדבר עליו עוד בהמשך

(The Yahoo! User Interface (YUI) framework)

 ודרכו אנו מעלים את סיפריית הAxe במלואה, אולם אם יש משהו לא תקין יצא לנו פלט שגיאה שהתפירה לא צלחה.

הפעלת פופ-אפ בJQuery – צד לקוח

הטמעת הספרייה JQuery הייתה יותר פשוטה מכיוון שהספרייה כבר מוטמעת במאגר הספריות של המודל.

למעשה לא היה צורך להטמיע באותו אופן שבו תפרנו את ספריית הAxe אלא פשוט להשתמש ביבוא של הספרייה ולהתחיל לעבוד בה.

החלק המסובך היה לקשר את הפופ-אפ שנפתח למשתמש לפלאגין שיצרנו.

היה צורך להכניס style קבוע עם הפלאגין לכל ערכת עיצוב וללא תלות בערכות עיצוב של המודל (שיעבוד באופן תקין על כל ערכה במודל ללא בעיות צבעים, בנק מחרוזת וכדו'), לכן לא היינו יכולים להישען על ערכת העיצוב הקיימת בפטל (או ערכת עיצוב הדיפולטיבית במודל ) על מנת שהפלאגין יהיה אוניברסלי.

ראשית דאגנו לעיצוב כתבנו קובץ CSS נפרד שמכיל את עיצוב התיבה, ובעזרת המטודה appendChild דחפנו את העיצוב בתגית Link לתוך האלמנט head.

עד כאן דאגנו לעיצוב, מכאן והלאה יש לדאוג לחלון התיבה הפיזי.

לקיחת המידע הקיים בתוך הEditor

לאחר חקירה רבה יצרנו תגית Div שקראנו לה contentDiv וקישרנו אותה לbody אשר קיים בכל עמוד.

בעיה נוספת שנוצרה היא איך לקחת את המידע שקיים בתוך הEditor.

מצאנו איך לשאוב את המידע הקיים בתוך הכתבן ולשלוח אותו לAxe, אבל נתקלנו בבעיה שזה היה עובד על Editor ספציפי במודל ולא על כל הEditors הקיימים במודל (למשל בפלאגין של לוח הודעות הייתה אפשרות לשלוח מידע אולם בכתבן של בוחן לא הייתה אפשרות כזאת).

לאחר מאמצים רבים מצאנו את התכונה המשותפת לכל הכתבנים במודל מסוג role הנקרא textbox.

document.querySelector('[role="textbox"]')

\*חשוב לציין שזאת הייתה בעיה שלא מצאנו לה פתרון גם בעזרת קהילת מודל העולמית ולאחר מאמצים רבים מצאנו לה פתרון באופן עצמאי אשר חידש לאנשי קהילת המודל.

מטודות של JQuery

המטודות שהשתמשנו בכפתור שלנו לצורך הUI/UX הינם:

* Open לדיאלוג שיפתח.
* Close לסגירת הדיאלוג ופתיחת AJAX שישלח לDB כמה שגיאות אותו יוזר עשה באתו זמן נתון.
* Prev & Next כפתורים למעבר בין השגיאות.
* כפתור Focus שיוכל לנמק את השגיאה ולהראות לנו בדיוק היכן התבצעה שגיאת הנגישות בEditor.

כתיבה לDB והצגתו - צד שרת

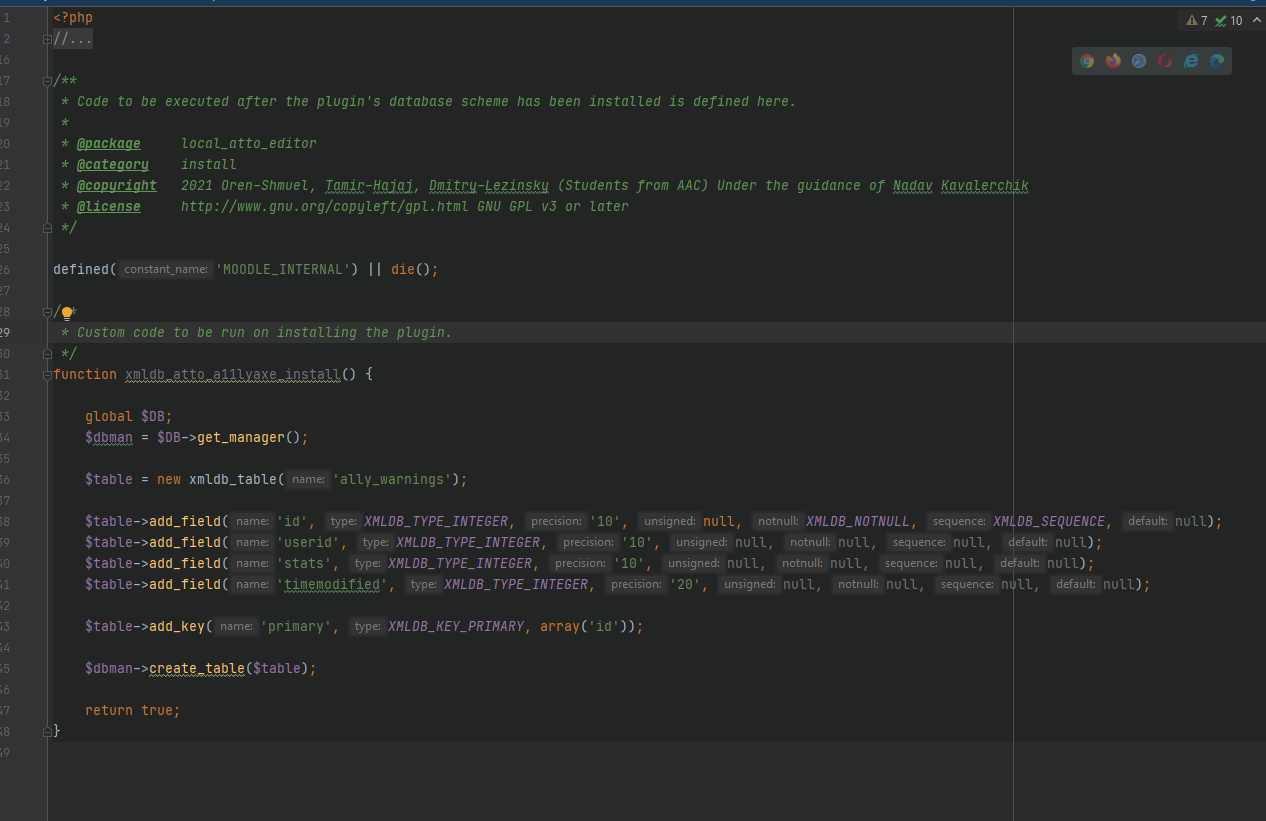
הקמת DB (install)

לצורך כתיבה לDB היה צורך בלהרים טבלה חדשה שממנה נוכל לכתוב ולייצא נתונים, לשם כך היה צורך לבצע התקנה של טבלה חדשה לפלאגין.

דבר זה נעשה בקובץ PHP אשר נקרא install.php והושם בתיקה db בתוך הפלאגין.

כאשר עושים התקנה מחודשת לפלאגין הקובץ מקבל invoke ומתקין טבלה עם הפרפיקס הרצוי שבחרנו כבר בconfig.php שלנו.

למעשה הקובץ נראה כך:

****

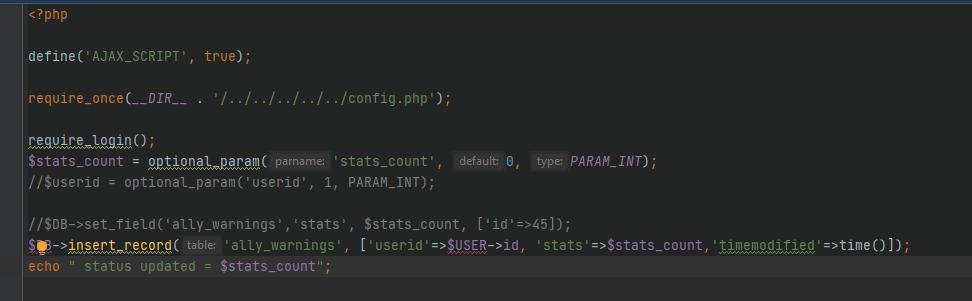
השורה הראשונה מחליטה שלא יהיה גישה לקובץ הנ"ל דרך ניווט מהמודל, דבר זה מונע כניסה לא רצויה לטבלה (שכל בן אדם יבוא ויוסיף דברים לטבלה כגון שדות או שאילתות וכדו').

אנו נשתמש בגלובאל DB לצורך בניה של טבלה חדשה כעת אנו נבצע מופע המשמש לביצוע פעולות ונעשה מניפולציות לצורך יצרת טבלה ושדות חדשים אליה:

בשורה 36 יצרנו טבלה חדשה הנקראת : ally\_warnings והכנסנו 4 שדות (id רץ, המספר המזהה של המשתמש, כמה שגיאות נגישות בוצעו באותו רגע, מתי הם בוצעו) לאחר מכן הפכנו את id למפתח ראשי של הטבלה ויצרנו אותה.

עדכון הDB

לאחר מכן יצרנו קובץ הנקרא: ajax\_update\_stats.php אשר משמש אותנו לכתיבה למסד הנתונים – הקובץ שואב את המידע מהAJAX שיצרנו את כמות השגיאות ופשוט כותב אותו לטבלה עם המידע של מי היוזר, ומתי זה קרה.



AJAX בצד לקוח

נחזור כעת לצד לקוח – לצורך מתן הנתונים הללו, רצינו להשתמש בטכניקה אשר לא מצריכה עדכון או ריענון. לשם כך הכרנו את AJAX.

היתרונות של AJAX:

* מעדכן את הדף ללא צורך בריענון.
* שולח נתונים לשרת ברקע.
* עובד יחד עם JQuery ו-PHP.

למעשה ביקשנו בקשה של POST והבאה לDB באופן של planeText – עידכון לקובץ ajax\_update\_stats.php עם המידע של כמה שגיאות בוצעו,

לאחר שיש תשובה מהשרת נכתוב "הצלחה" ואם לא נכתוב "נכשל" נראה כך: 

לאחר כל זה נעבור לחלק של מה כל ספרייה עושה ומה הקשר שלה אצלנו בקוד:

ספריות מבנה הקוד

GRUNT

Grunt הוא כלי שמבצע מיניפקציה לכל קבצי הפרויקט שלנו מאחר והמודל לא מוכן לקבל קבצי js ו-css באופן raw אלא רק לאחר קימפול.

למעשה המודל מוכן לקבל רק שני סוגים של קבצים:

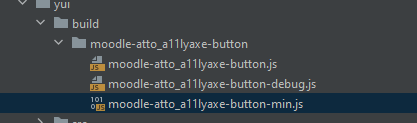
* 1. קובץ שנקרא min שזה מינפקציה של הjs הרגיל
  2. קובץ של דיבוג לצורך ניטור שגיאות בקובץ, מסתיים ב-debug

למעשה לאחר כל שינוי בקוד שלנו היה צורך לעשות "קימפול" לקבצי הjs שלנו אם היינו רוצים לראות את התוצאה בפועל, קימפול זה לוקח כ-2 דקות בכל הרצה שלו.

הפקודה של הקימפול נראית כך:

Grunt yui –force

היה צורך להשתמש בforce מאחר וסביבת עבודה שלנו phpStorm מאוד מדוקדקת והיא לא מוכנה לקמפל דברים של Es6 או אלמנטים חדשים שקיימים בjs (כמו למשל חריגת שורה, חוסר בנקודה פסיק, שימוש בגלובל אם לא נצרך, הצהרות של ספריות core שהוא לא מכיר וכן הלאה).



הייתה עוד אופציה שזה לשנות את קובץ grunt בהתאמה לסביבה שלנו אולם בהמלצת המנחה הוחלט שלא לעשות זאת כי זה כרוך בהבנה יותר עמוקה של מה המודל מבקש מהסביבת פיתוח ולעשות גישור בין הסביבה לחוקים של המודל לפי תאימות שקיימת, אלו דברים שלא קשורים לפרויקט שלנו הספציפי ולכן מנחה הפרויקט אמר לנו לא לעסוק בהם.

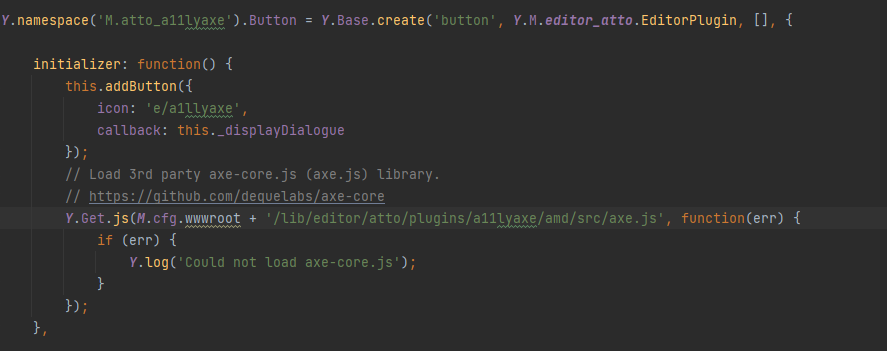
YUI

YUI היא ספריית JavaScript מורחבת ומודולרית. מספר מודולים זמינים עבור YUI המספקים מגוון רחב של פונקציונליות לבניית יישומי אינטרנט אינטראקטיביים עשירים תוך שימוש בטכניקות כגון Ajax.

מאז גרסה 2.4 YUI טוענת מספר מודולים סטנדרטיים והופכת אותם לזמינים במרחב השמות העולמי Y בהם השתמשנו במסגרת הפרויקט.

השימוש שלנו בפרויקט היה לקרוא לכל רכיבי המודל ע"י הקריאה לY.cfg (כאשר אם זה בphp אנו רושמים CFG)

עשינו שימוש בכך בהכנסת הספרייה של Axe ע"י ניתוב לCDN ולאחר מכן פנייה לספרייה בתוך התיקייה של הפלאגין.



- (Moodle docs והסברים של המנחים שלנו)

AMD

את כל הספריות שאנו מקימים (Axe ו-JQuery) אנו שמים בתיקיית מפתח שנקראת AMD.

התיקייה הנ"ל מעידה על השימוש בספריות שלנו במהלך הפרויקט. למעשה זה המקום שכל הספריות צריכות להישמר בכל פלאגין – ככה למעשה המפתחים הבאים אחרינו יודעים באיזה ספריות השתמשנו במהלך הפרויקט.

AMD זהו מודל של Javascript, Moodle תומך במודולי Javascript שנכתבו באמצעות ממשק (AMD).

(API Asynchronous Module Definition)

ה- API של הגדרת המודול האסינכרוני (AMD) מציין מנגנון להגדרת מודולים כך שניתן לטעון את המודול ואת התלות שלו באופן אסינכרוני. זה מתאים במיוחד לסביבת הדפדפן שבה טעינה סינכרונית של מודולים יוצרת בעיות ביצועים.

- (Moodle docs והסברים של המנחים שלנו)

**שלושת הספריות הללו היו נצרכות למהלך הפרויקט, להטמעת רכיבים וספריות שונות בJS או פלטפורמה אחרת. הם למעשה חלק מהLegacy של מודל ובלעדיהם אין אפשרות להטמיע משהו חיצוני במודל – השימוש בהם נעשה בעזרת המנחה שלנו ושאר משתתפים שעזרנו לנו במהלך הפרויקט.**

PHP & JS

למעשה השימוש העיקרי שלנו היה בשפות של JS וphp

כל הקריאות לדף שלנו מצד לקוח נעשו ע"י javaScript כגון יצירת div חדש בתוך הbody, הטמעת הdescription שקיבלנו מספריית הAxe וה-target לכל id (אם קיים או של תכונות אחרות שקיימות באלמנטים ששאבנו) והצגתם למשתמש בצורה נאותה ו/או פוקוס לרכיב הבעייתי.

בצד שרת השתמשנו בphp לקחת את המידע שהיוזר שלח, הכנסתו בתוך הDB והצגתו בתוך הדו"חות מתקדמים (מה שנקרא Configable Report) כל פעם שהוא לוחץ על סגירה.

JQuery

jQuery היא ספריית JavaScript הנתמכת על ידי דפדפנים שונים, שמטרתה להקל על כתיבת סקריפטים לצד הלקוח.

jQuery הופצה לראשונה בשנת 2006, וכיום משמשת בלמעלה מ-50% מתוך 10,000 האתרים הנצפים ביותר.

העבודה עם jQuery, נעשית באמצעות אובייקט מוגדר מראש הקרוי 'jQuery', כאשר לצורך נוחות, ניתן לסמן אותו באמצעות תו הדולר ($).

- (Wikipedia)

למעשה כל הדיאלוג שלנו נכתב בעזרת JQuery כולל הפנייה לצד שרת AJAX שנדבר עליה בהמשך.

למעשה היה צורך כמו שאמרנו להטמיע גם את JQuery וגם את JQueryui מטעמי Legacyלמרות שהשתמשנו רק ב JQuery.

מאחר שמערכת מודל עובד בערכות נושא שונות היה צורך לעשות ערכת נושא משלנו בסגנון JQuery שיהיה מוטמע ביחד עם הפלאגין ויוכל ללכת איתו לכל מקום, לשם כך הוספנו את הקובץ css: jquery-ui.css

בנוסף הוספנו עוד קובץ css (style.css) שיהיה אפשר להשתלט על הפוקוס – מטעמי נוחות הפרדנו את הקבצים, ושיהיה אפשר למתפתחים הבאים אחרינו להבין את הלוגיקה של הפלאגין.

AJAX

AJAX (ראשי תיבות של Asynchronous JavaScript And XML) היא טכניקה ליצירת יישומי רשת אינטראקטיביים המבוססים על קוד המורץ במסגרת דף HTML בודד, ולא כיישום מרובה דפים, כמקובל בסביבת ה-Web. מטרתה העיקרית של הטכניקה היא שיפור חוויית המשתמש והאצת מהירות הטעינה של דפי האינטרנט, מאחר שהיא מאפשרת לעדכן רק חלקים מבוקשים בדף האינטרנט, ללא צורך לטעון את הדף כולו מחדש במחשבו של המשתמש.

מטרה זו מושגת באמצעות יצירת תקשורת והחלפת מידע בין מחשב הלקוח לשרת דפי האינטרנט באמצעות קוד JavaScript. כלומר, האינטראקטיביות של יישומי AJAX מושגת באמצעות קוד המורץ בצד-הלקוח.

כאשר משתמשים ב-AJAX עדיין יש צורך בטכנולוגיית צד-שרת כמו PHP או ASP, אלא שבמקום לשלוח דף HTML שלם, נשלח ללקוח רק המידע, והלקוח מפרש את המידע ומציג אותו למשתמש.

- ( Wikipedia)

כאמור המידע שלנו בצד הלקוח היה כמות השגיאות שנעשו, את כל המידע הנוסף השתמשנו ב- M.CFG לשאיבת המידע בצד השרת.

Axe

זה למעשה ליבת הפרויקט שלנו, מנוע הAxe המכיל מעל ל200 חוקי נגישות באינטרנט, עשינו לו התאמה למודל והכננו אותו למערכת המודל (כל שאר המידע דיברנו כבר בפרקים קודמים, ראה בעיקר חלק 3).

Maria DB

שרת MariaDB הוא אחד ממאגרי המידע היחסיים הפתוחים בקוד פתוח. הוא מיוצר על ידי המפתחים המקוריים של MySQL ומובטח להישאר קוד פתוח. הוא חלק מרוב הצעות הענן וברירת המחדל ברוב ההפצות של לינוקס וגם אצלנו במודל.

- (Maria DB docs)

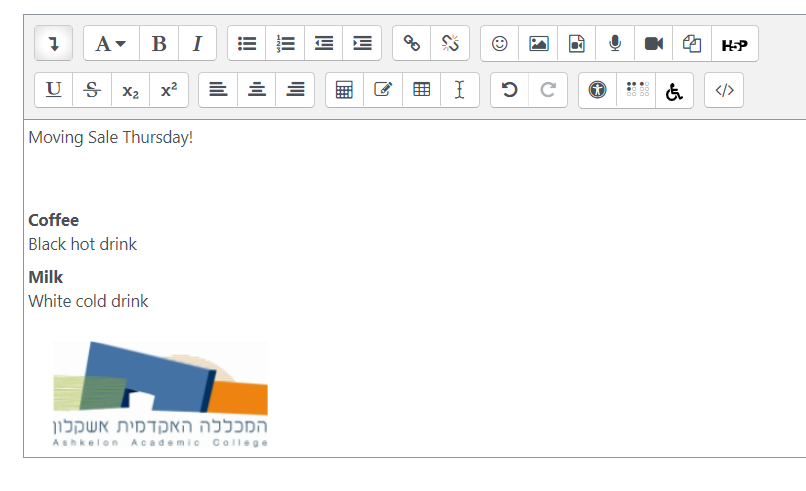
למעשה החלק העיקרי שלנו היה להכיר את מסד הנתונים בהתקנה ותהליך הקונפיגורציה של config.php וההכנה שלו לגירסה התואמת את הphpMyAdmin שלנו שקיבלנו מהשרת Wamp שעשינו.

בנוסף לכך הקמנו טבלה בקובץ install.php הנקראת "mdl\_ally\_warning" (כל הטבלאות של מודל מתחילות בפרפיקס הזה)

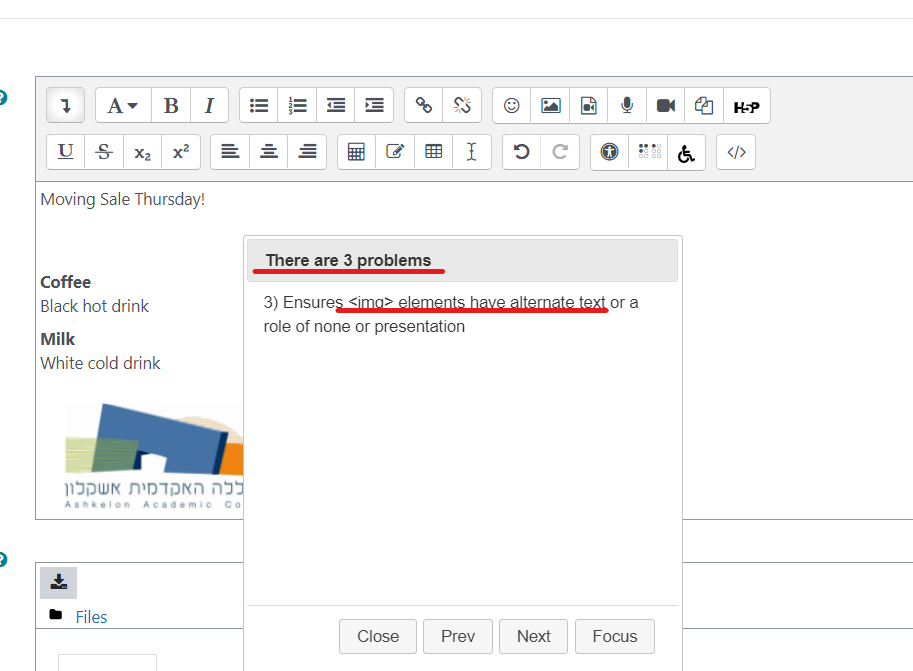
והעדכון לטבלה מתבצע בקובץ ajax\_update\_stats.php

הדגמה של הפלאגין:

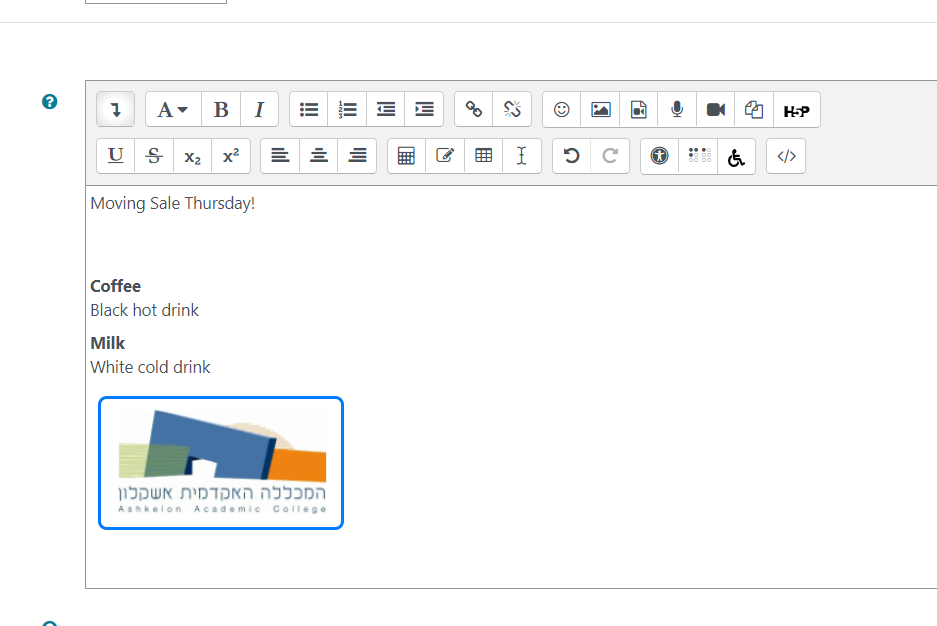
1. קיימות 3 בעיות נגישות בכתבן:



1. הפעלת הפלאגין ע"י לחיצה על הכפתור ופתיחת הדיאלוג עם שלושה בעיות נגישות





1. לחיצה על הפוקוס תגרום לסימון האלמנט הבעייתי וריחוף עם העכבר על האלמנט תמחק את הפוקוס שנוצר. 

חלוקת תפקידים:

בהנחיית המנחה נדב קבלרצ'יק ומכיוון שהפרויקט שלנו היה פרויקט שלבי, שבו יש צורך לסיים עם השלב הקודם בטרם ניתן להתחיל את השלב הבא, לא הייתה חלוקת תפקידים בינינו. כולנו נפגשנו במכון ויצמן ועבדנו על הפרויקט ביחד מול שרת לוקאלי אחד והעלאת שלבי ההתקדמות לAWS בסיום כל יום עבודה.

יש להבין שסביבת מודל היא לא פלטפורמה פשוטה של צד שרת ולקוח, אלא כל רכיב צריך לדעת לדבר עם כמה אלמנטים וצריך שכל מפתח מודל ידע להכיר את כל הרכיבים כולל צד לקוח , שרת ומסד נתונים בלעדי זה אין אפשרות להתקדם.

מצורף בעמוד הבא מסמך סיכום של המנחה:



**סיכום מיזם שיפור נגישות תוכן במערכת Moodle**

שלום רב,

בהמשך לדוא"ל קודם בנושא, בו הוצג פירוט תכולת המיזם: פיתוח רכיב (תוסף) "לשיפור נגישות הזנת תכנים במערכת Moodle" של המחלקה להוראת המדעים, אני מעוניין כעת לעדכן שהמיזם הגיע לסיומו באופן מוצלח ולשביעות רצוני, וגם זו של שאר הגורמים הרלוונטיים במחלקה.

התוצר הסופי הוא ממשק אשר מציג למורים אשר מזינים תוכן במערכת את המקומות בהם נוצר תוכן אשר איננו עומד בתקינה של חוקי נגישות, ומכוון אותם לפתרון הבעיות אשר התגלו.

לאורך המיזם, הסטודנטים עבדו, מצד אחד, בצורה עצמאית כדי לחקור ולתת פתרונות לאתגרים אשר עלו במהלך הפיתוח, ומצד שני, פעלו כצוות לשילוב הפיתוחים אשר בוצעו במקביל, כפי שמצופה מבוגרים של מוסד אקדמי אשר מכשיר סטודנטים לרכישת כישורים הנדרשים לעבודה ברוח התקופה, המאה ה 21.

ברמה הטכנית, הסטודנטים רכשו והפגינו ידע בשלל טכנולוגיות: תשתיות ענן, התקנה והגדרה של שרתים, פיתוח בשפת PHP, SQL וגם JAVASCRIPT, הכרות ושימוש בספריות JS שכיחות כגון JQUERY ו AXE. פיתוח צד-לקוח הכוללת תקשורת עם צד שרת יישומים ובסיס נתונים. למעשה, יכולות FullStack מלאות.

שמחתי מאוד ללוות את הצוות,

ומקווה להמשך שיתוף פעולה גם בעתיד.

בברכה,

נדב קבלרציק

ראש צוות פיתוח טכנולוגי,

המחלקה להוראת המדעים,

מכון ויצמן למדע