

# פרויקט גמר בניית אתרים

מסמך הנחיות ודגשים לפרויקט

## מפרט אפיון הפרויקט

### מבוא ומטרות

למידה עצמית היא **Skill** מאוד מרכזי במקצוע שלנו, ראוי להשתמש בו ככל האפשר במהלך הפרויקט. עם זאת, חשבו את הזמנים מראש (**עוד Skill** חשוב במקצוע שלנו) נא לסיים את הפרויקט ולהגישו בזמן. השתדלו לא להיגרר ללימוד ויישום עוד ועוד נושאים אם זה יפריע לכם להגיש את הפרויקט בזמן. נדרשת השקעה של כ-120 שעות פיתוח ויש עד 3 חודשים להגשה, המרצה/הרכזות יגדירו לכם את תאריך ההגשה לטובת הפרויקט. **תבנה אתר צד לקוח** עם **צד שרת** תומך בנושא שתבחר. על הפרויקט להיראות כמו אתר/ אפליקציה אמיתית עם תמונות ותוכן טקסטואלי **שלא כוללים lorem ipsum** וכדומה. כמו כן הפרויקט צריך להיות **רספונסיבי** ומתאים לכל גדלי המסך האפשריים.

### תיאור כללי:

פיתוח אפליקציה אינטרנטית הכוללת מערכת ניהול אתר המאפשרת **למנהל האתר** **לפרסם תוכן**. התוכן שיפורסם יהיה **זמין בחלקים שונים של האתר**. (לדוגמה אם מדובר בחנות אינטרנטית, לאחר שמוסיפים מוצר הוא צריך להופיע בדף הראשי או בדף מוצרים)

### דגשים מרכזיים:

- אתר מרכזי הכולל עמוד תצוגת תוכן.
- מערכת התחברות הכוללת גישה לממשק ניהול אתר.
- ממשק ניהול האתר יאפשר: הוספה, עריכה או מחיקה של תוכן.
- תוכן האתר יישמר בצד שרת.

### אופן הגשת הפרויקט:

את הפרויקט יש להעלות ל - git ללא תיקיות ה - node\_modules, ולשלוח במייל נפרד קובץ **ספר פרויקט/ מסמך אפיון** שיכלול קובץ טקסט **ReadMe** שמסביר על הפרויקט, תכולתו, הפונקציונאליות ודרך ההתממשקות עמו. **אם יש** קבצי .env או config עם פרטים אישיים יש לצרף גם אותם למייל, אחרת יש להעלות אותם ל - git. יש לרשום במייל שם מלא של התלמיד, שם הקורס, מספר הקורס ומס' טלפון שניתן להשיג את הסטודנט בעזרתו.

### הערה כללית

לא ניתן להגיש פרויקט סוף מודול כפרויקט גמר.

### דרישות כלליות

- יש לשמור על קוד נקי ומסודר: לנקות console.log וקטעי קוד שהפכו להערות.

- מומלץ לבנות קוד שמספר סיפור ולתת לפונקציות ולמשתנים שמות משמעותיים.
- יש לחלק את הפרויקט למודולים לפי נושאים
- יש לשים את כל הקשור לעיצוב בקבצי css, את התמונות בתיקיית images וכו'.
- כמו כן יש להקפיד על המוסכמות לכתיבת קוד.
- עיצוב הוא חלק בלתי נפרד מהצגת הפרויקט והיכולות של הפיתוח. גם אם אינך מעצב באופי, הקפד על עיצוב נקי ורספוניסבי לגדלים שונים של מסכים!

חשוב לזכור שפרויקט זה יהיה חלק מתיק העבודות שלכם וייצג אותך בכבוד מול מעסיקים פוטנציאליים ולכן יש לשמור על אסתטיקה של קוד ועיצוב

## דרישות טכנולוגיות צד לקוח

1. **עיצוב ורספונסיביות** מומלץ להשתמש בספריית bootstrap או Material Design
2. **קובץ העיצוב הראשי** אם קובץ העיצוב (css) הוא מעל 100 שורות, יש לחלק אותו לקבצים נפרדים לפי הנושאים השונים. לצורך העניין מומלץ להשתמש בספריית scss
3. **אייקונים** מומלץ להשתמש באייקונים לדוגמה מספריית fontawsom או google material
4. **דף כניסה** צריך לכלול כותרת ראשית, כותרת משנית, טקסט תמונה שיתאמו לאופי האתר/ האפליקציה. אם מדובר באתר של חנות אינטרנטית כלשהי, יש להציג בדף הפתיחה שדה חיפוש ולפחות שלושה כרטיסי מוצר. מדף הפתיחה צריך להיות ברור לאיזה סוג של אתר/ אפליקציה הגענו וצריך להיות מעוצב בצורה כזאת שתזמין את הגולש להירשם.
5. **תפריט ניווט** על האתר/ אפליקציה להכיל תפריט ניווט דינאמי שמשותף לכל דפי האתר
6. **Footer** על האתר / אפליקציה להכיל footer עם לוגו, זכויות יוצרים ואמצעי ליצור קשר עם האתר. במידת הצורך ניתן להוסיף גם בתפריט הניווט, קישורים למדיה חברתית או כל דבר אחר שיתאים לאזור זה באפליקציה/ אתר
7. **נגישות** יש לשים את שם האפליקציה בתגית ה - title בקובץ ה - index הראשי, וכן תמונה/ לוגו ב - link:favicon. כל תמונה חייבת לכלול את האטריבוט alt עם כיתוב שיתאר את התמונה.
8. **דף אודות** יש ליצור דף אודות בו תספקו הסבר מעמיק על האתר ודרך ההתממשקות עמו.
9. **הירשמות והתחברות** על ממשק צד לקוח להציג דף התחברות ודף הרשמה הכוללים וולידציות על שדות הטפסים השונים. צריך להיות חיווי מתחת לשדה הרלוונטי אם המשתמש עומד או לא עומד בדרישות השדה או תוך כדי ההקלדה או on focus change לשדה אחר. יש לאפשר שליחה של טופס אך ורק לאחר שכל שדות החובה מלאים. בלחיצה על כפתור השלח בטופס, יש לעדכן את הגולש בהצלחה או כישלון שליחת הנתונים. כדי להשתמש לצורך העניין בספריית react-tostify או בספרייה דומה. מומלץ לאפשר התחברות גם באמצעות חשבון Google, Fasebook, וכו.
10. **Regex** יש להשתמש ב - regex לשדה הסיסמה בטפסים, שיחייב הכנסת סיסמה עם לפחות אות אחת גדולה ואות אחת קטנה באנגלית, לפחות ארבעה מספרים ולפחות סימן מיוחד מבין הסימנים הבאים (\*\_!@%\$#&~) ושכל הסיסמה תהיה לפחות בת 8 תווים. כמו כן יש להשתמש ב - regex לשדה של טלפון שיחייב הכנסת מס' טלפון תקני אם אפשרות לרווח או מקף אמצעי (-) ( לאחר שנים או שלושת המספרים הראשונים. וכן regex על שדה המייל שיחייב הכנסת מייל

- תקני. יש אפשרות ליצור פונקציות בצד שרת לצורך העניין או להשתמש בספריית jio-browser.
11. **Token** לאחר התחברות יש ליצור token עם ערך מוצפן תוך שימוש בספריית jwt. ה - token צריך להישמר ב - localStorage ולעזור בקביעת סטאטוס המשתמש ואם הוא מחובר או לא. רק בתוך ה - token יכולים להיות פרטים על הלקוח באופן מוצפן.
12. **Crud** לאחר התחברות יש לאפשר למשתמש את פעולות ה - crud, קרי: קריאה, יצירה, עדכון ומחיקה של תוכן. התוכן שיוצרים צריך להיות זמין בחלקים שונים של האתר. לדוגמה אם מדובר בחנות אינטרנטית, לאחר שמוסיפים מוצר הוא צריך להופיע בדף הראשי או בדף מוצרים. יש להשתמש בספריית react-tostrify או בספרייה דומה, כדי לעדכן את הגולש בהצלחה/ כישלון ביצוע פעולות ה - crud.
13. **מועדפים/ הכנסה לסל קניות** יש לתת אפשרות לגולש לשמור תוכן (כרטיס/ מוצר/משתמש וכ"ו) במועדפים. יש לתת חיווי ויזואלי לכך שהתוכן הוא מועדף על ידי הגולש. יש לשמור את העדפות הלקוח ברשומת התוכן במאגר המידע כך שלא משנה מאיזה מכשיר המשתמש ייכנס לאתר/ אפליקציה התוכן שהוא העדיף ימשיך להיות מועדף. יש ליצור דף של תוכן מועדף, בו הגולש יוכל לראות את כל הפרטים שהוא סימן כמועדפים ואם ירצה יוכל ולהסיר מהדף פריטים מועדפים
14. **דף פרטי תוכן** בלחיצה על כרטיס/ משתמש/ תוכן, הגולש יעבור לדף דינאמי בו יינתנו פרטים נוספים על פריט התוכן עליו לחץ הגולש
15. **שדה חיפוש** יש ליצור שדה חיפוש לתוכן (כרטיס/ מוצר/משתמש וכ"ו)
16. **הרשאות** יש לאפשר יצירה של לפחות שני סוגי משתמשים, כאשר הראשון הוא משתמש רגיל והשני הוא admin. רק משתמש שמוגדר כ - admin יוכל ליצור, לערוך ולמחוק תוכן, בעוד שמשתמש רגיל יוכל רק לראות או לסמן תוכן כמועדף עליו.
17. **קריאות http** יש לבצע קריאות http לצד שרת מצד לקוח ובאמצעותם ולשלוח ולקבל מידע מהשרת. מומלץ להשתמש לצורך כך ספריית axios. יש להשתמש במנגנון try & catch בקריאות אסינכרוניות לצד שרת במקרה ומשתמשים בפונקציות אסינכרוניות, או לחלופין במנגנון .then().catch(). עם בוחרים להשתמש ב - promises וזאת על מנת שהקוד לא יישבר במקרה ותחזור שגיאה קריטית מהשרת.
18. **ארכיטקטורה** יש לשמור על סדר הגיוני ומקובל בתעשייה של קבצים. יש להשתמש בתיקיית services, על הקוד להיות נקי וקריא, עם חלוקה נכונה לתיקיות וקומפוננטות.
19. **Console** יש להקפיד על עבודה נכונה עם ה - console. על הקונסול להיות נקי מהערות אזהרה, שגיאות ותוכן, כך שיהיה ניתן לראות בקלות שגיאות קריטיות מהשרת.
20. **סינון תוכן** יש לתת לגולש אפשרות לסנן את התוכן המוצג בדף מסוים לפי פרמטרים שונים

## בנוס לצד לקוח

1. **Logout** יש לאפשר לאפליקציה/ אתר לנתק את המשתמש במידה ולא השתמש באתר/ אפליקציה במשך יותר מ -4 שעות
2. **הגבלת בקשות** יש להגביל את מספר הקריאות לשרת שמשתמש יכול לבצע ב - 24 שעות לצורך

הגנה על השרת ממתקפה שנועדה להאט/ להקריס אותו.

3. **ממשק ניהול משתמשים** דף ניהול משתמשים שיציג את מספר המשתמשים הרשומים במאגר המידע באתר ויציג בטבלה את פרטיהם. בדף זה יהיה ניתן למחוק משתמשים או לשנות את הסטאטוס שלהם ממשתמש רגיל admin.
4. **ניהול הזמנות/ מועדפים** דף שיראה לאדמין את התוכן לפי כמות האנשים שהגדירו את אותו התוכן כמועדף עליהם.
5. **ניהול מלאי** במקרה של חנות אינטרנטית, דף לניהול מלאי של חנות. וברגע שמוצר מסוים נגמר מהמלאי יש לתת חיווי לגולש על כך שהמוצר אזל.
6. **תמונת פרופיל משתמש** עדכון פרטי משתמש עם תמונת פרופיל שתועלה לתוך תיקייה במחשב עליו נמצא הפרויקט
7. **עדכון סיסמה** יש ליצור דף שיאפשר למשתמש ששכח את הסיסמה שלו להחליף סיסמה וזאת רק לאחר וידוא שאכן מדובר במשתמש שמנסה לשנות את סיסמתו ולא האקר. לצורך העניין מומלץ לשלוח מייל למשתמש עם לינק לדף לשינוי הסיסמה.

## דרישות טכנולוגיות צד שרת

1. **package.json** בקובץ package.json יש לשים ב - devDependencies את nodemon, וכן שיהיה בתוך מפתח ה - "main" עם שם הקובץ להפעלת האפליקציה.
2. **האזנה לבקשות HTTP** יש לבנות ממשק rest api שמאפשר קבלת בקשות, ליצירה, עריכה, הצגה ומחיקת מידע ממאגר המידע בהתאם לבקשות צד לקוח. לצורך העניין מומלץ להשתמש בספריית express
3. **אותנטיקציה** יש להעביר את הבקשה תהליך של אותנטיקציה על מנת לוודא שאכן הגולש הוא זה ששלח את הבקשה ולא האקר שמנסה לפרוץ למאגר המידע דרך השרת.
4. **אותוריזציה** יש לאפשר בצד שרת רק למשתמש מחובר ומוגדר כ - admin לבצע הוספה, מחיקה או עריכה של מידע ממאגר המידע.
5. **מאגר מידע** את המידע בפרויקט יש לשמור בבסיס הנתונים MongoDB או mySql בצורה לוקלית או על ענן. כמו כן יש לאפשר הוספה, עריכה ומחיקה של פרטים ממאגר המידע. בקובץ שאתם מגישים חייב להיות קובץ ה - config במידה ושמתם בו את המפתחות להתחברות עם שרת הענן
6. **ולידציות צד שרת** יש לעשות ולידציות צד שרת עם ספריית Joi או ספרייה דומה, ובמקרה של שגיאות יש ולעצור את הפונקציות בטרם שליחת האובייקט לוולידציה של mongoos ושמירה במאגר המידע.
7. **Routes & Models** יש לחלק את הקוד למודולים לשמור על קוד נקי וקריא
8. **Logger** יש להשתמש בספרייה לניהול בקשות http כדוגמת morgan או לחליפין ליצור logger משלכם שידפיס בקונסול קריאות מצד לקוח לצד שרת
9. **הערות** על שמות המשתנים והפונקציות להיות הגיוניים ושיספרו סיפור על הקוד. במידת הצורך

---

יש להוסיף הערות תמציתיות למתכנתים במידה ויש פונקציות מורכבות או שיש דף עם ריבוי פונקציות בתוכו.

---