חפיפת React

1. יש לעבור על כל הסרטונים כאן <https://scrimba.com/learn/learnreact>

ניתן לדלג על סרטונים שבהם הוא חוזר על דברים שכבר דיבר עליהם בעבר

1. לחקור על styled components ועל css modules, דרכי השימוש בשניהם ולבחור אחד מהם לעיצוב הפרויקט שלכם.
2. משימת חפיפה (להודיע לחופף כשהגעתם לכאן):

הערות למשימה:

1. **מטלת החפיפה חדשה וייתכן ופספסתי מקרי קצה או דברים כלליים שהכרחיים עבור המשימה, אז לא להתבייש לשאול הכול מול החופף.**
2. לא צריך להילחץ מהמשימות, אתם לא נמדדים על זמן ועדיף לוודא בכל סעיף שהכול תקין לפני שעוברים לסעיף הבא.
3. מצופה מכם שכל דבר שאתם לא בטוחים מה דרך המימוש הכי טובה או כלים שיכולים להקל עליכם מאוד את העבודה (חלק מהם מומלצים בסעיפים עצמם איפה שרלוונטי וחלק לא) שתחקרו באינטרנט או במידת הצורך תשאלו את החופף/הצוות.
4. לא לשכוח לטפל במקרי קצה, תפיסת שגיאות (אם זה מהשרת אז הדפסת/הצגת שגיאה משלכם או מה שחוזר מהשרת) וסדר כללי של הקוד מבחינת קומפוננטות, פונקציות, כללי תכנות, להכיר כל דבר שאתם כותבים או לוקחים מהאינטרנט, ובכלל להתכונן לשאלות בעל פה על הקוד/דרכי מימוש/דברים אחרים שלמדתם בחפיפה.
5. במקומות שאתם יכולים לעצב בעצמכם אתם לא נמדדים על העיצוב עצמו אלא על הלוגיקה והרעיון, ובמקומות שאתם צריכים לעקוב אחרי עיצוב קבוע חשוב לנסות לעשות כמה שיותר דומה, ואם יש קשיים במהלך הדרך להעלות מול החופף.
6. בדרך כלל פנייה לשרת\API תהיה מעין קופסא שחורה ולא תדעו מה עושים עם המידע שלכם, במטלה כאן הקוד שבשרת פתוח בפניכם ואתם יכולים להסתכל על הקוד ולהבין את הflow הכללי של המערכת (מבלי לשנות את הקוד).
7. החפיפה כתובה בלשון זכר אבל מכוונת לשני המינים.

המון בהצלחה!!!!

עליכם להקים אפליקציית React שתשמש כצד לקוח, ולהוריד את צד השרת מהכתובת הבאה:

<https://github.com/Lirontal/ReactHafifa>

נרצה להקים אתר שיציג לנו חיזוי מזג אוויר לשבוע הקרוב.

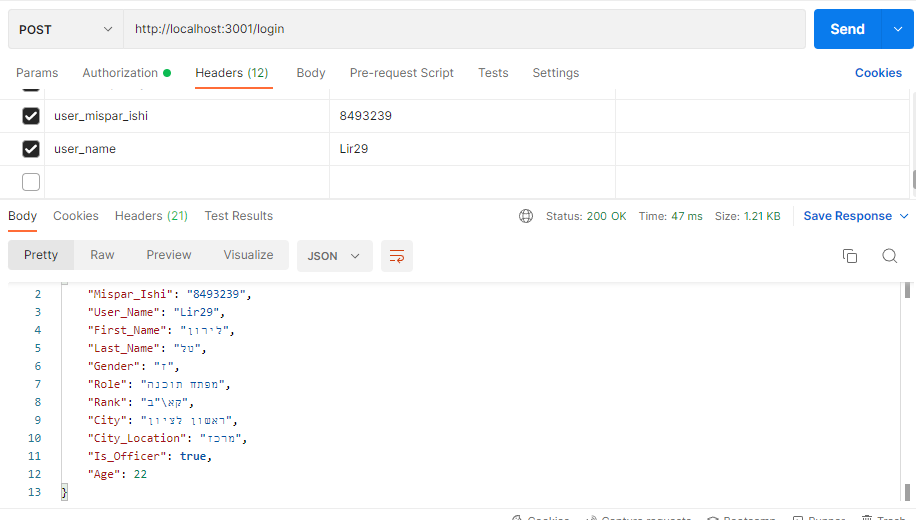
1. בעליית האתר נקים דף הזדהות הכולל שני inputים, אחד לשם משתמש, והשני לסיסמא (מספר אישי), ובנוסף כפתור לאישור הטופס. הטקסט שכתוב בinput של הסיסמא יהיה מוסתר (כוכביות). בעת לחיצה על הכפתור תהיה בדיקת ולידציה על הנתונים (שם המשתמש חייב להיות אך ורק אותיות באנגלית, לפחות אות אחת גדולה וקטנה ומקסימום 3 מספרים, מומלץ להשתמש בRegex), ולאחר הולידציה הנתונים ישלחו לשרת לאימות.

הפנייה לשרת תהיה לכתובת

<http://localhost:3001/login>

בקשת POST

צריך להוסיף את הheaders הבאים לפנייה הזאת ולכל פנייה אחרת לשרת בלבד (לא לצרף לAPI החיצוני שתראו בהמשך): 



במידה וההזדהות נכשלה (כל response שונה מ200) יש להציג בפרונט הודעת שגיאה ולא להכניס למערכת, במידה והצליחה יש לשמור בlocalStorage (לחקור) את שם המשתמש והסיסמא ובנוסף להכניס אותו למערכת. יש לדאוג לבדיקה של הlocalStorage כל פעם שהמערכת עולה ואם הסשן כבר שמור שם צריך לשלוח אותם ישר שוב לשרת לאימות ואז לעבור למערכת עצמה ללא דף ההתחברות. הקריאה מהשרת תחזיר את המידע המלא של המשתמש.

לגבי עיצוב מסך זה אפשר איך שרוצים (לא להשקיע יותר מדי או לדאוג אם זה יוצא יפה או לא), רק שהכול יהיה ממורכז באמצע.

1. לאתר יהיו כמה routes (ניתן להשתמש בreact-router-dom), ועל מנת להקל עליכם את ניהול המידע יש צורך להשתמש בhook בשם useContext

המסך הראשי של האפליקציה יהיה בראוט /home ואמור להיראות כך:



למעט השינויים הבאים:

כל האפליקציה תהיה בעברית, למעט החלק העליון שיהיה select מבין אופציות שנגדיר בהמשך, ולידו יהיה כפתור שבתוכו יהיה אייקון של זכוכית מגדלת

התמונה מאחורה וכל התמונות עצמן ניתן לבחור חופשי מהאינטרנט

במקום הימים בשבוע יש לכתוב: " היום, מחר, בעוד יומיים, בעוד 3 ימים, בעוד 4 ימים" בהתאמה

מעל הselect יהיה כתוב "שלום" ואז השם הפרטי ושם המשפחה של המשתמש שמחובר.

לפני עליית האפליקציה לאחר ההזדהות (כלומר בכל ראוט שננחת בו לאחר מסך ההזדהות וולידציית המשתמש נקרא לשרת על מנת להשיג את שמות הערים שניתן לחפש, עד שהנתונים ייטענו נציג אינדיקציה שהנתונים בטעינה ולא נציג את האתר עצמו (למשל loader), יש לדאוג לטפל במצב של שגיאה בהשגת המידע.

קריאה לשרת <http://localhost:3001/getAllCities>

GET

את האפשרויות שנקבל נטען לתוך הselect והבחירה הדיפולטיבית שלו תהיה Jerusalem. כל פעם שנלחץ על הכפתור עם זכוכית המגדלת נשלח את הערך שכרגע נבחר לשרת ל http://localhost:3001/cities/cityName (GET), השרת יתרגם את שם העיר לכתובות long וlat, יחזיר ללקוח ואז תתבצע פנייה לapi <https://openweathermap.org/api/one-call-api> באמצעות המפתח הבא: 6f11fa9760902e1597265ad205f05d2c (בעולם אידיאלי היינו שומרים אותו בצד שרת ולא חושפים אותו בצד לקוח), המפתח צריך להיות שמור במשתנה סביבה.

1. עבור המספר היומי של מזג האוויר, יש להשתמש בממוצע של temp.max וtemp.eve
2. עבור האייקונים של מזג האוויר, יש לבחור אייקון בהתאם ללוגיקה והעדיפות הבאים:

הtemp.day גדול מ29: שמש

הclouds גדול מ20: שמש עם ענן

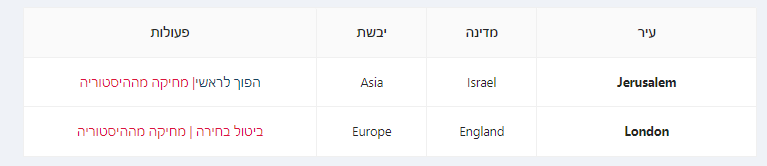
Pop גדול מ40: גשם

אחרת: קשת בענן

1. בתוך כל חלון של יום נרצה להציג את כמות החלקים ביום שבהן ממש חם לעומת הטמפרטורות בפועל. נשווה את האובייקט של feels\_like עם האובייקט temp ונראה כמה פעמים הטמפ' המורגשת גדולה מ(הטמפ' בפועל +1 ) באותו הזמן. אם זה קורה פעם אחת ביום ספציפי יש לצבוע את הרקע של החלון של היום באפור, אם פעמיים ביום ספציפי בכתום, ואם יותר מפעמיים באדום.
2. לאחר חלק זה יש לעבור על חפיפת הgit, לבצע שם את המשימות הראשונות ולאחר מכן להעלות את צד הלקוח שהקמתם כאן לrepository ייעודי בgithub (יש ליצור משתמש שישמש אתכם בהמשך לפרויקטים של הצוות). צריך לדאוג שמשתני הסביבה ותיקיית node modules לא יעלו לrepo.
3. עכשיו יש לעבור לבראנצ' בשם mission7, יש לשמור את תוצאות החיפוש במערך, כך שתמיד ישמרו מקסימום ה5 חיפושים האחרונים (משתנה סביבה). בסוף המשימה להעלות את הקוד לבראנצ' mission7 ולאחר מכן להעלות למאסטר.
4. נחזור לבראנצ' מאסטר, משם נייצר בראנצ בשם header ובו נייצר header שיופיע בכל route לאחר ההתחברות ויאפשר לנו לעבור ביניהם, ויראה כך, יש לדאוג שהheader תמיד יהיה מקובע למעלה:

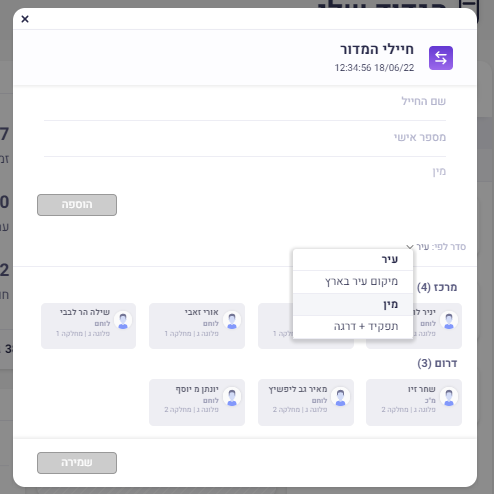


1. מבראנצ' header יש לצאת לבראנצ' בשם mission9 ובו נוסיף בheader טאב נוסף בשם היסטוריה ולידו מקף עם כמות החיפושים של מזג האוויר ששמורים במערך, בלחיצה עליו נעבור לroute חדש ובו תציגו טבלה עם כל החיפושים האחרונים ששמרתם במערך, בטבלה שאמורה להכיל את הנתונים הבאים (העיצוב לבחירתכם)



במקרה זה העיר האחרונה שחיפשנו היא לונדון, יש אפשרות לבטל את הבחירה בה כך שאם נחזור לroute המרכזי לא יוצגו הנתונים של לונדון, ומחיקה מההיסטוריה שתוציא את החיפוש ממערך החיפושים וגם תבטל את הבחירה בעיר אם היא כרגע בחורה.

במקרה של ירושלים, יש אפשרות להפוך לראשי, בלחיצה על זה ירושלים תהפוך לתוצאה שנציג את התוצאות של החיפוש שלה ונחזור אוטומטית לroute המרכזי (יש לשים לב לא לבצע פנייה מחדש לapi אלא להשתמש בנתונים שכבר קיבלתם)

1. נחזור לבראנצ' header (כלומר שהקוד שלנו לא יכיל את התכולות של mission9, ובו יש ליצור route חדש בשם mador. בטעינת העמוד תתבצע קריאה לשרת <http://localhost:3001/getAllSoldiers> GET שממנה תקבלו את כל נתוני החיילים במדור מהשרת. לאחר שהטעינה הצליחה יש להציג כפתור במרכז המסך שהטקסט שלו יהיה "פתיחת חלון", ובלחיצה עליו ייפתח החלון הבא: 

מטרת הpopup היא להציג את החיילים שכרגע במדור בכמה אופנים, ולתת אפשרות מחיקה של חיילים מסוימים והוספת חדשים

בחלק של הכותרת יש תאריך המייצג את התאריך הנוכחי, יש לדאוג שהוא יהיה דינאמי ויתעדכן כמו השעון האמיתי בכל שנייה.

בחלק העליון צריך להיות 3 מקומות של טקסט (שם החייל ומספר אישי כמו שמופיע בתמונה, ויש להוסיף "שם משתמש" מתחת) ואחד בחירת מין זכר/נקבה, לאחר הזנת כל הדברים כפתור ההוספה יצבע ויהיה ניתן ללחוץ עליו על מנת להוסיף את החייל החדש באופן זמני לרשימת החיילים (שאר הנתונים של החיילים שלא אפשרנו בחירה כמו עיר מגורים/גיל ניתן להזין משהו דיפולטיבית/רנדומלי כל דבר שתרצו, רק לוודא שאם מדובר למשל בגיל אז שיהיה מספר ולא מחרוזת). יש לוודא ששם המשתמש מורכב מאותיות באנגלית ולפחות אות אחת גדולה וקטנה ומקסימום 3 מספרים וניתן להשתמש באותו הRegex שהשתמשתם בו קודם לפני שהחייל נוסף לרשימת החיילים.

בחלק התחתון נציג את החיילים עצמם, נאפשר אפשרות סידור לפי הקטגוריות הבאות: עיר, מיקום עיר בארץ, מין, ותפקיד+דרגה. במקרה של תפקיד+דרגה בגלל שמדובר בקיבוץ לפי 2 תכונות ולא לפי אחד, מספיק שחייל לא יהיה תואם לחייל אחר באחד מהדברים והוא יהיה שייך לקטגוריה נפרדת.

הכותרות של הקיבוצים: במין במקום ז/נ יהיה כתוב זכר/נקבה, בתפקיד+דרגה יש לכתוב את התפקיד והדרגה מופרדים ע"י פסיקים

מבחינת בחירת הסדר לפי עצמו יש לשים לב לעיצוב הנדרש, האופציה שכרגע נבחרה (עיר) מסומנת בבולד, האופציה שעוברים כרגע עליה עם העכבר גם בבולט והרקע של המלבן שלה נהיה אפרפר.

הסדר הפנימי של החיילים בקטגוריה (למשל בקטגוריית עיר – רמת גן) יהיה תמיד לפי א"ב של השם הפרטי + המשפחה).

אורך המלבן של החיילים יהיה קבוע בפיקסלים, וכמות חיילים בשורה תהיה בהתאם לכמה מקום יש בcontainer בצורה אוטומטית

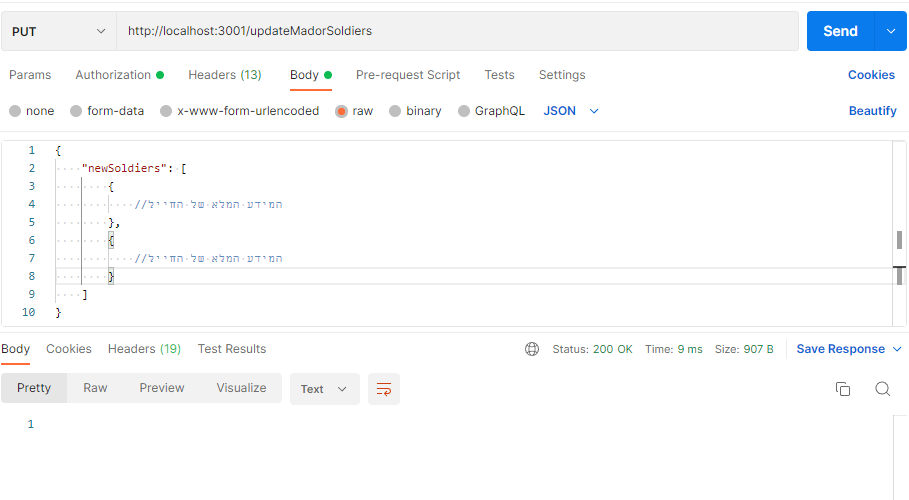
אייקון החיילים (זכר/נקבה) יהיה בהתאם למין שלהם. בשורה הראשונה יהיה את שם החייל, בשנייה את התפקיד + הגיל שלו מופרדים ע"י פסיק, ובשלישית את התפקיד שלו והדרגה שלו מופרדים ע"י פסיק.

יש לדאוג לגלילה פנימית רק בתוך החלק של החיילים, כלומר אם אין מקום להראות את כל החיילים אז הגלילה תתבצע רק בתוך הקונטיינר הרלוונטי ולא בתוך כל הpopup.

בלחיצה על מלבן של חייל הוא יסומן בצורה כלשהי וישאר מסומן עד ללחיצה נוספת עליו.

ליד כפתור השמירה למטה יהיו שלושה כפתורים נוספים בשמות "בחר הכול" ו"נקה הכול" ו"מחיקת מסומנים", בלחיצה על בחר הכול כל החיילים יהפכו למסומנים, ובלחיצה על נקה הכל כל החיילים יהפכו ללא מסומנים. במידה ויש לפחות חייל אחד שכרגע בחור כפתור "מחיקת מסומנים" יהפוך לפעיל ובלחיצה עליו החייל ימחק בצד לקוח בלבד מרשימת החיילים במדור. בלחיצה על שמירה נשלח לשרת את כל החיילים שכרגע במדור והם יהפכו להיות החיילים המעודכנים (לא קיימת ולידציה בצד שרת ולכן כל דבר שתשלחו לשם יעודכן, אז אם בטעות עדכנתם את החיילים בצורה לא נכונה יכולים להריץ מחדש את השרת) . רק במידה והשמירה הצליחה הpopup ייסגר.

הפנייה לשרת לשמירת החיילים:



בפנייה הזאת לפעמים, בצורה רנדומלית, הבקשה תיפול.

בכל לחיצה מחוץ לpopup או על כפתור הx הpopup ייסגר והשינויים לא ישמרו.

במידה וחייל הוא קצין (is officer === true), האייקון שלו יוצג כך (אינדיקציית קצין):



מומלץ להיעזר בעיצוב שהצוות שלנו קיבל למסך דומה בקודקוד ובעמודים סמוכים בזפלין:

<https://app.zeplin.io/project/5f57890a1390103e0582a685/screen/61e5378cebccc9834eea2ce2>

את האייקונים של החיילים ניתן להוריד מכאן: <https://zpl.io/29WBOEp>

יש להעלות את התכולה הזאת לבראנצ בשם mission10

1. יש לחזור שוב לבראנצ header ובו לייצר כפתור בheader שיהיה צמוד לצד שמאל שלו ובו יהיה כתוב "התנתקות", בלחיצה עליו המשתמש יתנתק מהמערכת (לא לשכוח לטפל בlocalStorage) ולאחר מכן לרפרש את העמוד כך שנחזור לדף הlogin. בסוף להעלות את התכולה הזאת לבראנצ בשם mission11
2. יש למרגג' את תכולות כל הבראנצים האחרונים (9-11) לבראנצ סופי בשם final. (באמצעות pull request ו/או github cli ) <https://cli.github.com/manual/gh_pr_create>

**בהצלחה 😊 😊 😊**