

## תרגיל ראשון - Web Client

### להגשה עד לתאריך 24.4

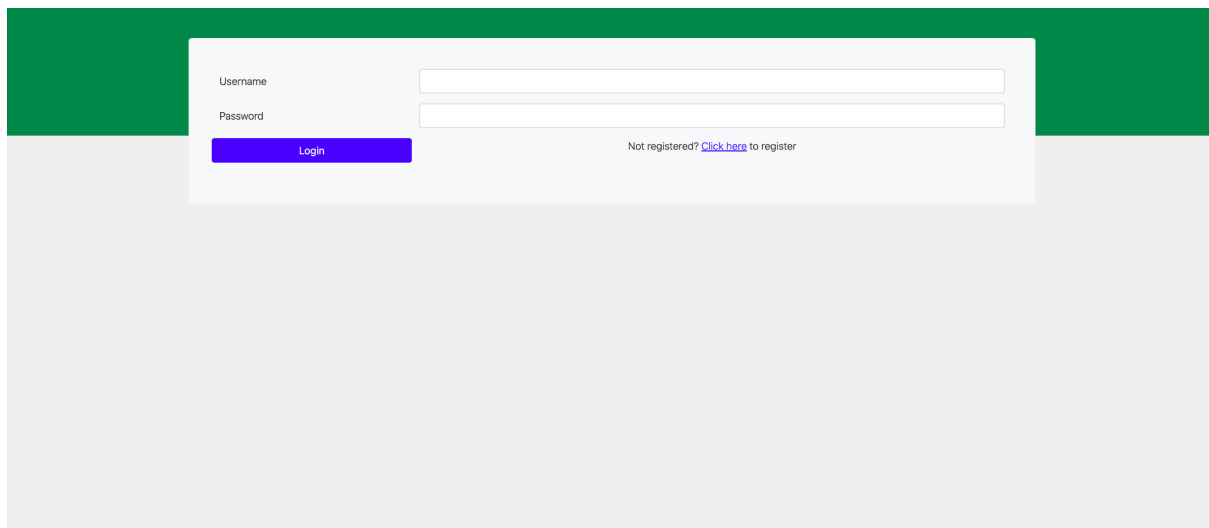
במהלך הקורס נבנה פרויקט מתגלגל שיורכב מ-4 שלבים. התוצר הסופי יהיה אפליקציית צ'אטים בעלת שלושה סוגי client ושרת.

בתרגיל זה עליכם לכתוב Web Client בעזרת HTML וJavaScript ולעצב בעזרת CSS.

#### יש לייצר שלושה מסכים:

- מסך הרשמה.
- מסך התחברות - שם משתמש וסיסמא.
- מסך צ'אטים - רשימת הצ'אטים והתוכן שלהם.

בעמוד הראשי יוצג **מסך התחברות** פשוט, בו יש להזין שם משתמש וסיסמא או מעבר למסך ההרשמה. דוגמא למסך כזה מוצגת בתמונה הבאה. (עם עיצוב בסיסי ביותר - כמו כל הדוגמאות כאן).



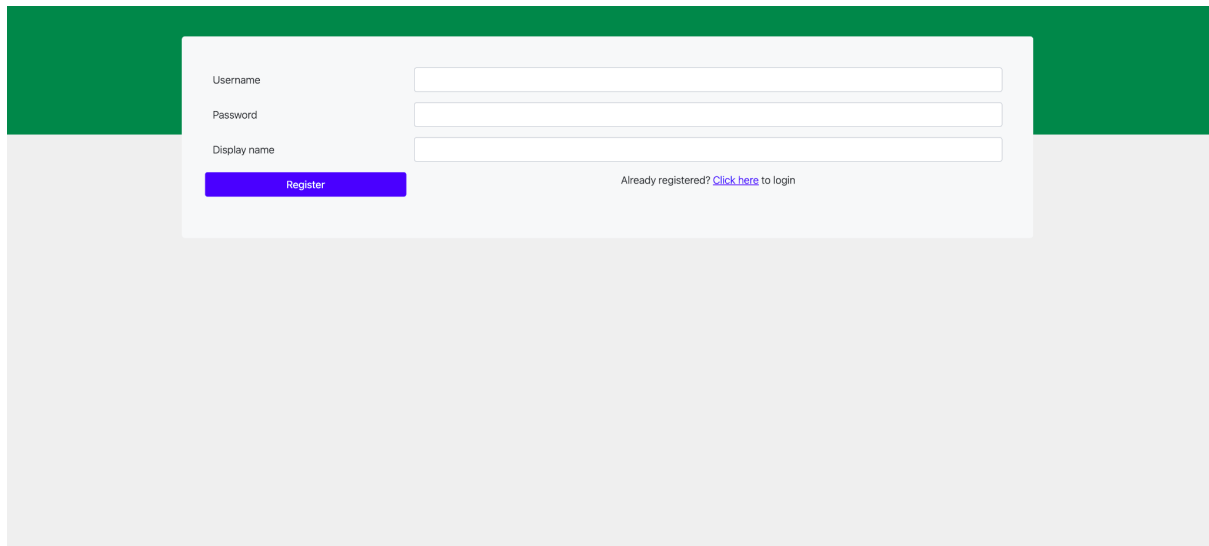
The image shows a login form with a green header and a light gray body. The form is centered and contains two input fields for 'Username' and 'Password'. Below the password field is a blue 'Login' button. To the right of the button, there is a link that says 'Not registered? [Click here](#) to register'.

כדי להתחבר בהצלחה - יש לממש קוד שבודק אם ההתחברות חוקית, כלומר: האם אכן נרשם בעבר משתמש כזה. יש להוסיף בדיקות תקינות קלט פשוטות נוספות, כמו בדיקה שהשדות אינם ריקים. **שימו לב** - שמכיוון שבתרגיל זה עליכם לפתח קליינט בלבד ללא שרת, הבדיקה האם נרשם כזה משתמש בעבר תהיה כנגד רשימה Hard Coded אשר תהיה בקוד ה JavaScript שלכם. במילים אחרות, עליכם ליצור מערך קבוע של אובייקטים בקוד שלכם, ולבדוק כנגדו האם פרטי המשתמש שהוזנו נכונים. בתרגיל הבא הבדיקה תהיה מול השרת שאתם תכתבו.

לאחר התחברות מוצלחת - יוצג מסך הצ'אטים של המשתמש שהתחבר.

**במסך ההרשמה** נקבל מקום להזין שם משתמש, כינוי (Nickname) לתצוגה, תמונה, סיסמא ווידוא סיסמא. יש לבצע ולידציות על שם המשתמש והסיסמא (חוקים בסיסיים), לדוגמא: סיסמא לא ריקה, עם תווים ומספרים.

מבחינת נראות, מסך ההרשמה דומה למסך ההתחברות, רק עם שדות מתאימים. דוגמא למסך ההרשמה מוצגת בתמונה הבאה.



### מסך הצ'אטים:

**ברשימת הצ'אטים (החלק השמאלי במסך)** יופיע עבור כל צ'אט (כלומר, עבור כל משתמש שאיתו אנחנו מדברים):

תמונה של המשתמש, הכינוי שלו, ההודעה האחרונה שנשלחה והתאריך/שעה שבה הגיעה.

**בעת לחיצה על צ'אט**, יוצגו בחלק השני של המסך, ההתכתבות (כלומר ההודעות) עם אותו נמען. הממשק צריך לתמוך בהוספת הודעות חדשות להתכתבות, ולכן יש לשים שדה קלט וכפתור מתאימים, ולממש לוגיקה מתאימה.

התכתבות יכולה להכיל הודעות מ4 סוגים: **טקסט, תמונה, וידאו וקול**, ויש לממש קוד מתאים לכל הסוגים הללו.

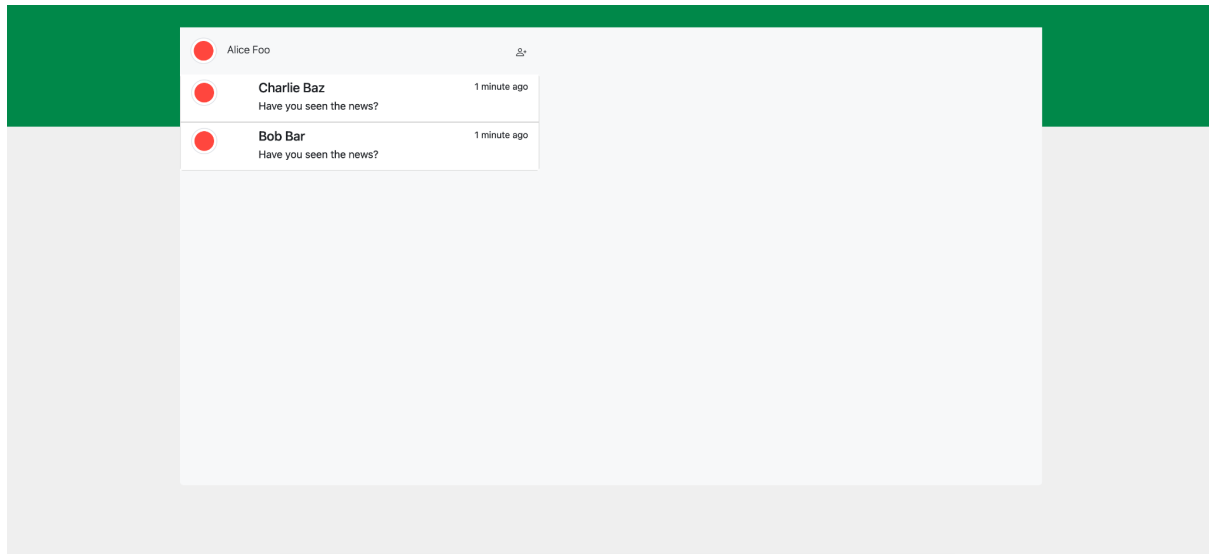
**כפתור הוספה של נמען חדש.** יש להציג חלון מודאלי (פופאפ) שיאפשר הזנה של שדה (של שם המשתמש של הנמען) וכפתור הוספה.

בקוד שאתם מגישים, צריכים להיות 5 נמענים לפחות (ברשימת הנמענים), וההתכתבות עם נמענים צריכה להכיל הודעות מכל הסוגים שהוגדרו לעיל (טקסט, קול וכו').

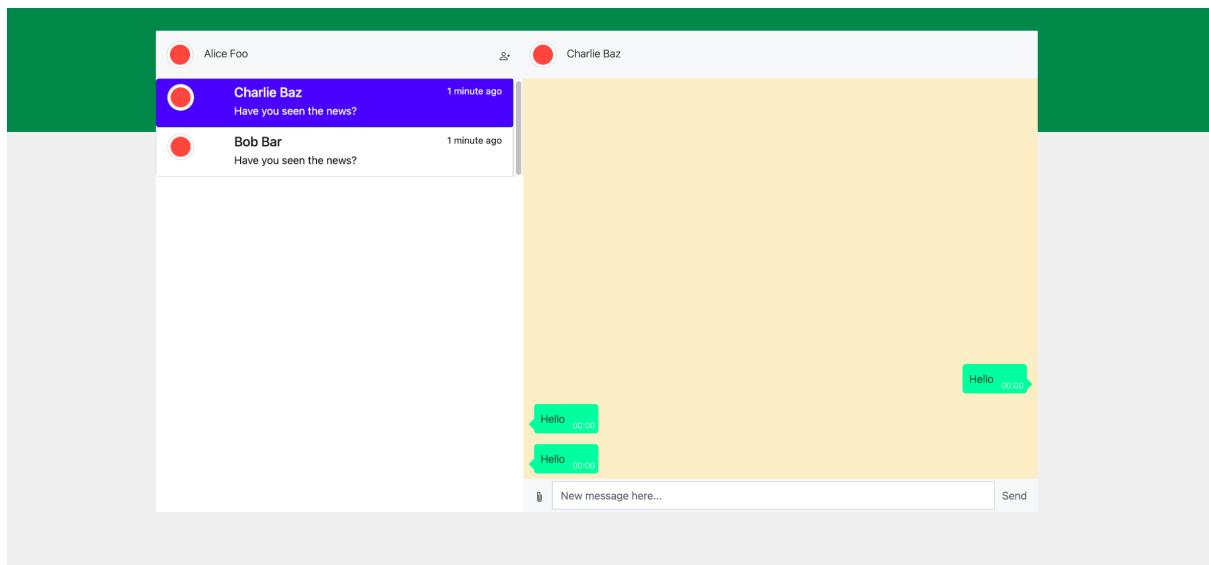
גם כן, בתרגיל זה כל המידע מנוהל במערכי אובייקטים hard coded ב Javascript. שימו לב כי בעת הוספת מידע חדש (נמען חדש, הודעה חדשה וכו'), על המידע להישמר במערכים הנ"ל. זה אומר שאם מזינים הודעה חדשה, ההודעה החדשה נשמרת, ובמידה ועוברים לנמען אחר וחוזרים לנמען הקודם, תוצג ההודעה החדשה. (עם זאת, במידה ויבוצע רפרוש של העמוד, יופיע רק המידע המקורי ולא המידע שהוזן).

ראו דוגמאות בעמודים הבאים.

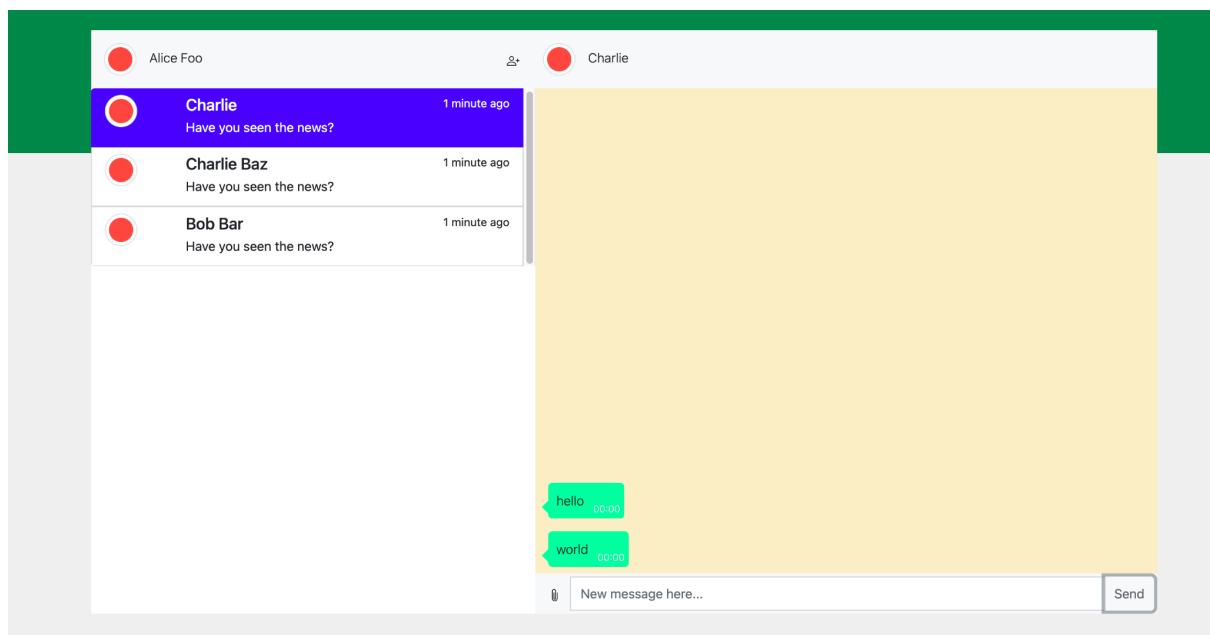
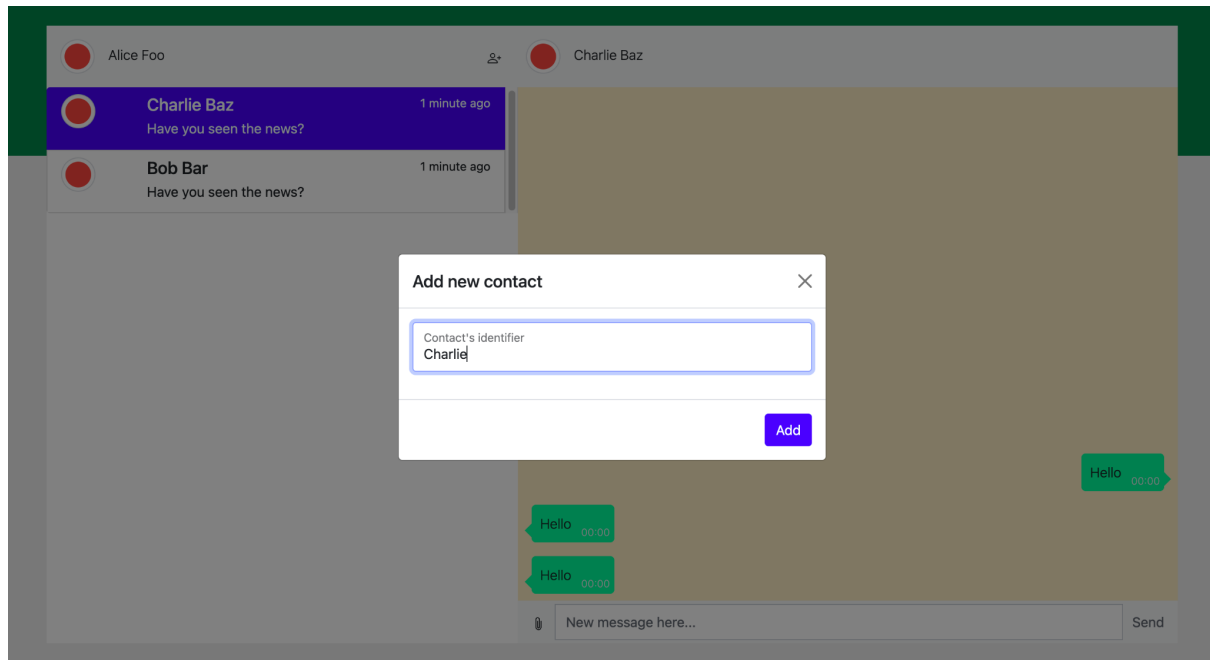
דוגמא למסך צ'אט, ללא צ'אט נבחר, מוצגת בתמונה הבאה.



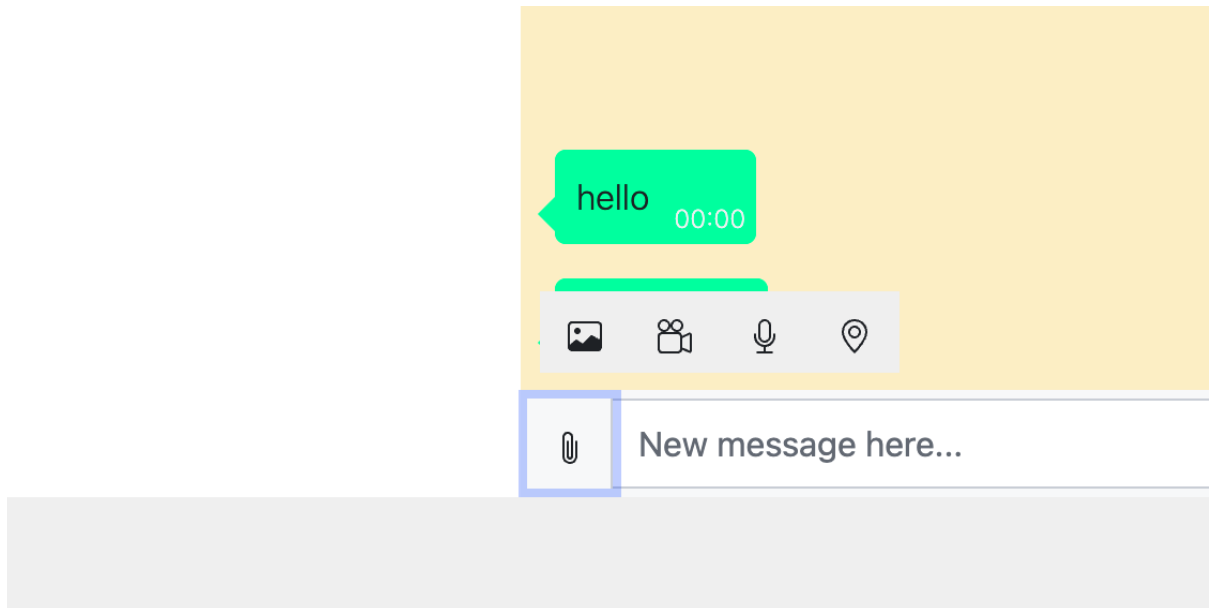
דוגמא למסך צ'אט, בעל צ'אט נבחר, מוצגת בתמונה הבאה.



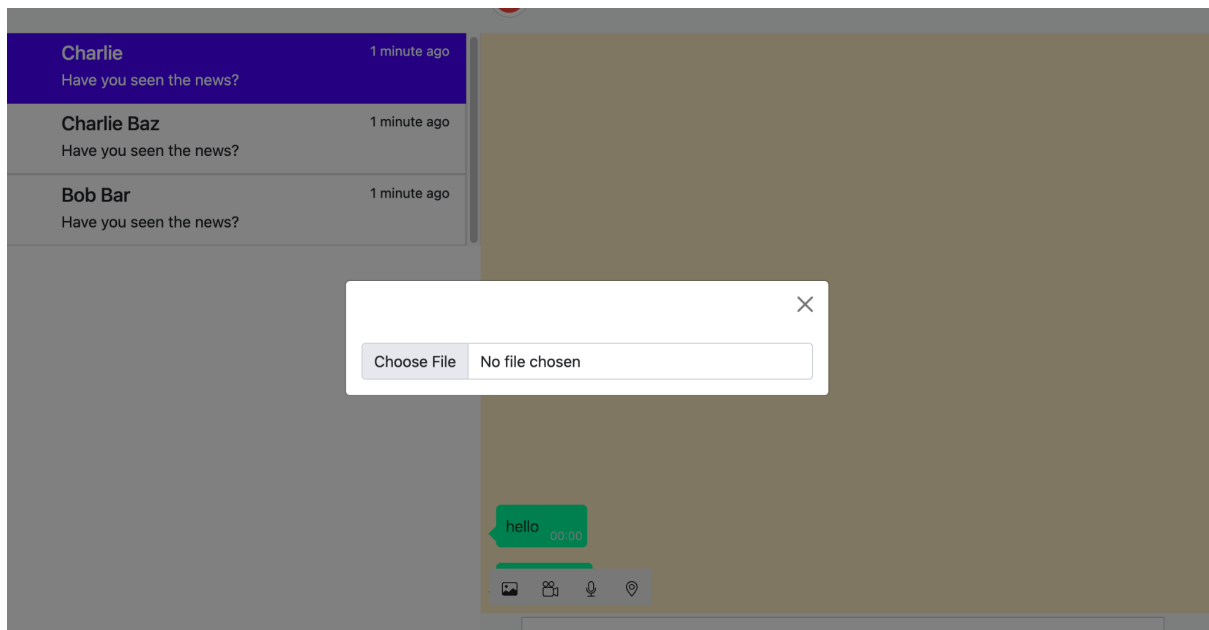
דוגמא לקטקס הוספת נמען + חלון הצ'אטים עם הנמען החדש, מוצגת בתמונות הבאות.

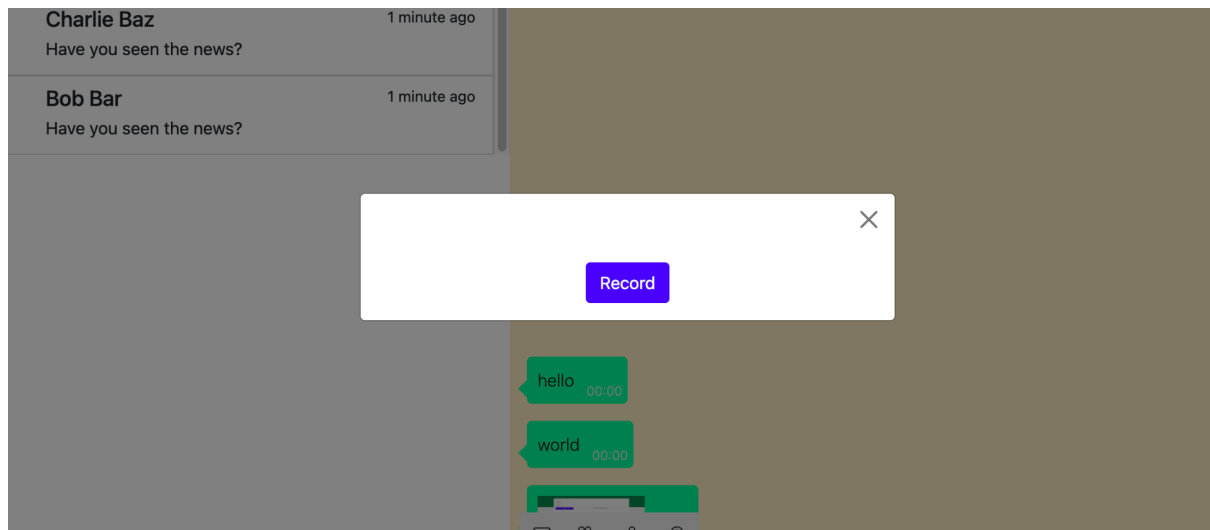


דוגמא לשדה הטקסט + כפתור המאפשר הוספת סוגי הודעות שהם לא טקסט, מוצגת בתמונה הבאה.

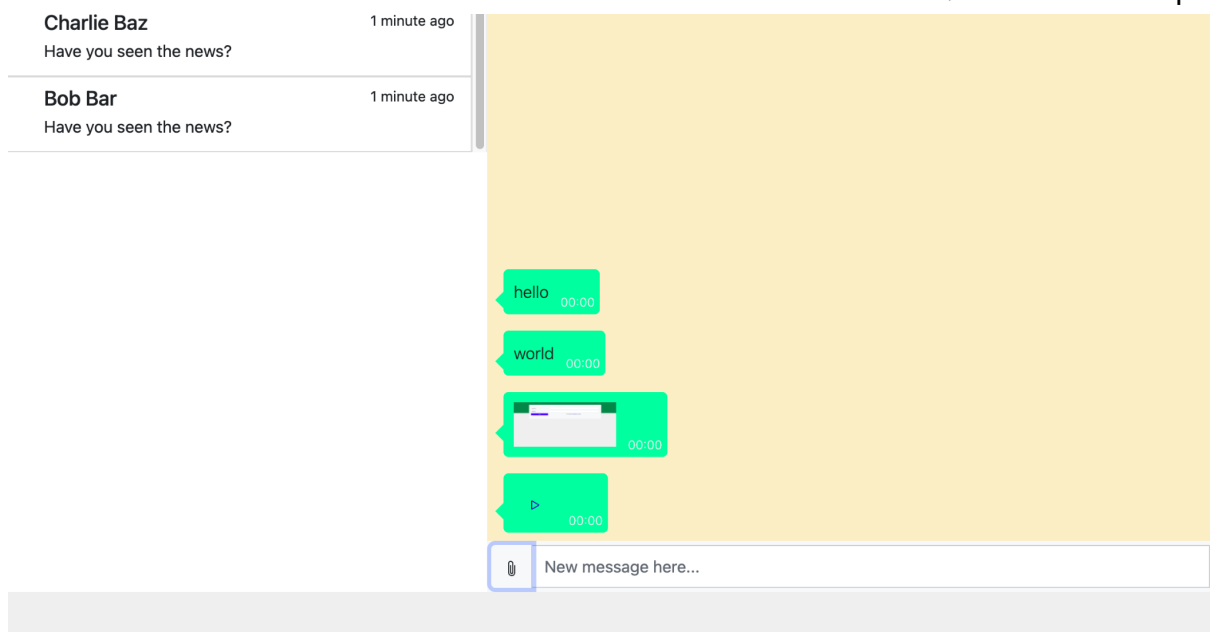


חלונות הוספת קובץ והקלטת וידאו (כדי לשלוח הודעה מסוג 'קובץ' ומסוג 'הקלטה') מוצגים בתמונות הבאות.





## מסך הצ'אט המציג הודעות מסוגים שונים.



### הערות:

נדרש עיצוב 'סביר'. יינתן בונוס על עיצוב מעולה. ירדו נקודות על עיצוב מינימלי. (ניתן לקבל יותר מ 100 בתרגיל זה בעקבות בונוס אך לא ניתן לקבל יותר מ 100 בציון התרגול הסופי. במידה וציון התרגול הסופי יהיה מעל 100, הציון יעוגל ל 100) שימו לב שכל הדוגמאות שנתנו הם ברמת עיצוב נמוכה (אך מספיקה).

**התמונות הנ"ל הינן להמחשה. אנו מעודדים אתכם להגדיל ראש ולהגיש ממשק משודרג אשר מכיל מעבר למינימום הנדרש, אך חייב לכלול את המינימום שהוגדר.**

מותר להשתמש ב HTML, CSS, BOOTSTRAP, JAVASCRIPT, JQUERY, REACT.  
מותר לעבוד בזוגות או שלשות (לא ביחידים).

## חובה לעבוד עם GIT.

חובה לעבוד ממשתמשים שונים - כלומר, כל סטודנט עובד מהמשתמש שלו. למשל, לא ניתן לעבוד זוג על אותו מחשב. שימו לב שניתן לראות ב-GIT כמה כל אחד תרם לפיתוח הקוד. דבר זה יבדק ויובא לידי ביטוי בציון.

## מי שעובד בשלשות:

חייב לעבוד ב-REACT. (מומלץ גם לזוגות אך לא חובה). כלומר, לא מספיק רק לשים את ה-html ושאר הקוד בסביבה של REACT, אלא ממש להשתמש בתשתית. כלומר, לחלק את הקוד ל-components. הדבר ייבדק על ידי הבודק. חובה להשתמש **לפחות** ב-useState, useRef. חובה להשתמש ב-router של REACT כך שכל הדפים הם חלק מאפליקציה אחת, ואין "ריפרוש" של העמוד במעבר בין הדפים.

## הגשה:

יש להגיש את הלינק לrepo שלכם במערכת המודל + שמות המגישים ותעודות הזהות שלכם בקובץ טקסט. אנא הוסיפו לrepo ב-GIT, קובץ README עם הסברים על אופן הרצת הקוד. במידה ואתם עובדים עם REACT אנא העלו ל-GIT את קוד ה-source (הקבצים שנמצאים בתיקיית src) ולא את האפליקציה הבנויה. הrepo צריך להיות private. בהמשך נספק לכם אימייל שאותו תוסיפו לrepo כדי שנוכל לבדוק את הקוד שלכם ללא סכנה להעתקות. שימו לב, אין לשנות את הrepo לאחר ההגשה. כל שינוי לאחר ההגשה ייחשב כאיחור.

## איחורים:

מותר להגיש את התרגיל עד 5 ימי איחור מעבר לתאריך ההגשה המופיע בתחילת התרגיל. כל יום איחור מתבטא בהורדה של 10 נק' נוספות בציון התרגיל. אין הבדל בהגשה באיחור של שניה או 23 שעות ו59 דק'. יום שישי ושבת נחשבים יום אחד. כלומר, אין הבדל בהגשה במהלך יום שישי או במהלך שבת. **לא ניתן להגיש מעבר לימי האיחור כלל. הגשה שכזאת, או שינוי ב repo לאחר ימי האיחור יגרמו לאי בדיקת התרגיל.**

אנו מעודדים את הגשת כל התרגילים במועדם. עם זאת, לרשותכם חמישה ימי איחור (בסה"כ) במהלך הסמסטר אותם תוכלו לנצל. כלומר, בסוף הסמסטר נחזיר עד 50 נקודות. [לדוגמא: תוכלו להגיש באיחור של 3 ימים את התרגיל הראשון, באיחור של יומיים את התרגיל השלישי ובאיחור של יומיים את הרביעי, ובסך הכל ירדו לכם 20 נקודות בלבד]. אין בונוס למי שלא מנצל את ימי האיחור הללו. אין צורך להודיע לנו על מימוש ימי האיחור, הדבר ייבדק אוטו' בסוף הסמסטר.

**שימו לב**, לא ניתן יהיה להגיש את התרגיל האחרון לאחר סוף הסמסטר (גם אם נותרו לכם ימי איחור). כלומר ההגשה לא תתקבל בכלל בניגוד לשאר העבודות בהן יהיה עונש של 10 נקודות על כל יום איחור (מעבר ל5 ימי האיחור שקיבלתם).

**עבודה עצמית בלבד. הנושא ייבדק ויאכף.**

בהצלחה!

