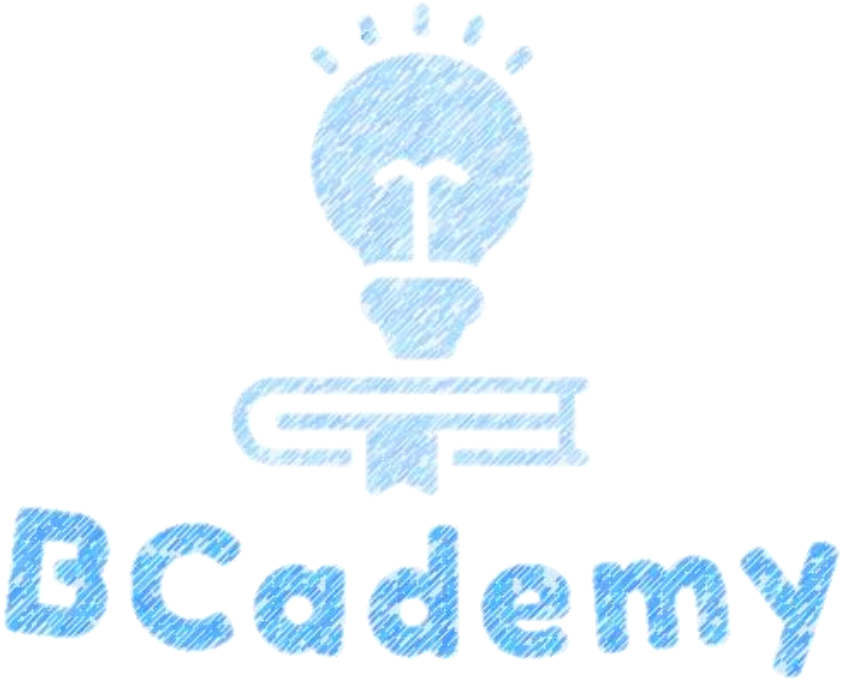
תיק פרויקט גמר במגמת סייבר –

BCademy



נושא העבודה: תיק פרויקט גמר, אפליקציית BCademy

שם המגיש: גפן חג'ג'
ת.ז: 322996323
בית הספר: תיכון בליך, רמת גן
מורים: אלדד קפיטולניק ואורי לוי

מועד הגשה: מאי 2019

קישור לקוד וצילומי מסך בגיט-האב: https://github.com/GefenHajaj/FinalProject

תוכן עניינים וראשי פרקים

פרק א' – BCademy – ייזום

נושא הפרויקט:
אפליקציה שתאפשר לתלמידים ללמוד לבגרויות ולמבחנים בכל הנושאים באופן קל ונוח.

מה הפרויקט עושה ולמי הוא מועיל?
כאמור, התוכנה שלי, שתהיה אפליקציה, תעזור לכל התלמידים ללמוד לכל המבחנים שלהם במהלך השנה באמצעות פיצ'רים שונים שתציע, כגון:

- בסיס הנתונים של התוכנה יכיל את כל החומר של כל מקצועות הלמידה, באופן שמחולק לנושאים קטנים. תלמיד יוכל לבחור מקצוע, כמה זמן יש לו ללמוד למבחן ומה החומר (מרשימת הנושאים המוצעת) והאפליקציה תעזור לו לתכנן את הלמידה ומה ללמוד מתי.
- תלמיד יוכל לבחור לתרגל למבחן מסוים שעומד להגיע – המערכת תציע לתלמיד לקרוא את החומר או (לפני מבחן למשל) לתרגל את החומר ולענות על שאלון קצר על החומר שקיים במבחן. כך, לפני מבחן התלמיד יוכל לשבת עם הטלפון ולענות על שאלות קצרות שירעננו את הזיכרון ויתרגלו את החומר.
- המערכת תהווה פלטפורמה לשיתוף חומרי לימוד, סיכומים, מצגות וכדומה. לכל תלמיד או מורה יהיה פרופיל, אליו יוכל להעלות חומרי לימוד באופן משותף עם כולם (או באופן פרטי) בנושא מסוים.

הסיבה לבחירת הפרויקט הזה:
לתלמידים רבים יש קושי לקראת מבחנים ללקט את כל חומר הלימוד ולהשיג אותו באופן מסודר ונוח לקריאה וללמידה. בנוסף, רבים מהתלמידים לא בטוחים כיצד לתרגל את החומר הנלמד ואם רוצים בכך – לא תמיד בטוחים מאין לתרגל את החומר! האפליקציה שלי תרכז את כל חומר הלימוד לתיכוניסטים ולתלמידי היסודי במקום אחד ונוח, כך שעל מנת ללמוד למבחן, לקבל סיכום, לתרגל ולקבל פידבק על הידע שלהם הם לא יצטרכו לחפש בשום מקום אחר חוץ מהאפליקציה!

מה האתגר בביצוע הפרויקט?
ראשית, אני רוצה לפתח את האפליקציה כך שתעבוד בפלטפורמת האנדרואיד. ואמנם, אני גם רוצה שהאפליקציה שלי תעבוד על פלטפורמת ה-iOS של אפל. בכל זאת, לי יש אייפון. לכן, אשתמש בפלטפורמה חדשה לגמרי של גוגל שבדיוק בתקופה שבה התחלתי לעבוד (חודש דצמבר של 2018) יצאה גרסה 1.0 שלה – Flutter. מעולם לא פיתחתי אפליקציה בסדר גודל כזה, ובמיוחד לא בפלטפורמה חדשה לגמרי (משמעות הדבר שבקושי יש מידע עליה באינטרנט).

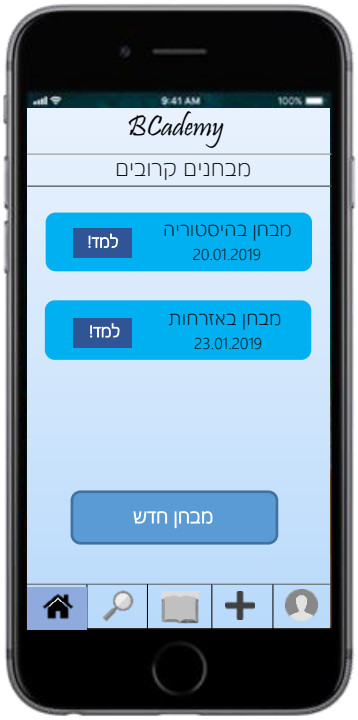
בנוסף, יש כאן צורך לפתח יכולת לתמוך במשתמשים רבים, כאשר כל אחד יכול להתחבר, להעלות סיכומים וחומרי לימוד, להוריד סיכומים וחומרי לימוד ולהצביע על רמת הסיכומים שאחרים העלו – ממש רשת חברתית ללמידה. לכן, יש כאן צורך לדעת לנהל בסיס נתונים שיכלול חומר רב שאני מכניס לו (כל חומר הלימוד בנושאים מסוימים + שאלות על כל נושא לתרגול) וחומר ומידע שמשתמשים אחרים יכניסו.

בנוסף, אתגר שהוא פחות תכנותי, יש כאן צורך לאסוף את כל החומר שקיים בכל המקצועות ולהעביר אותו לדיגיטל באופן מסודר ונוח, למפות אותו לנושאים קטנים כך שהתלמידים יוכלו לבחור בדיוק על מה הם רוצים ללמוד ולהכין שאלות על כל נושא. בשלבים מתקדמים יותר אף ניתן יהיה לפנות למשרד החינוך ולקבל סיוע בליקוט חומרי הלימוד ושאלות עליהם, בנוסף לאישור מטעמם על כך שהאפליקציה והחומר הניתן בה נכון ומועיל ואף לזכות בתמיכה מטעמם באופן ארצי, כך שהאפליקציה תזכה לחשיפה בארץ!

פרק ב' – BCademy - אפיון

א. פירוט המערכת

הפרויקט שנקרא BCademy עוסק בתחום שעולה על סדר היום של מאות אלפי בני נוער וסטודנטים על בסיס יומיומי. נושא שגורם לרבים לתסכול, בלבול ולעתים אף לחרדה. הנושא שאני מדבר עליו הוא כמובן בגרויות ומבחנים, ובאופן ספציפי יותר – הלמידה אליהם וארגון החומר.
האפליקציה שלי, BCademy, תאפשר לכל תלמיד להתכונן למבחנים הבאים שלו באופן קל ונוח. היא תציע מספר פיצ'רים (יפורט בפרק הבא) ותהיה מורכבת ממספר מסכים שונים (יפורטו מיד) שיאפשרו חוויית משתמש זורמת ונוחה, תוך למידה מהנה, חווייתית, יעילה וחסכונית בזמן!



חוויית המשתמש תהיה דומה מאוד לחוויה של אפליקציות מודרניות.
5 מסכים שונים, בתחתית המסך בר ניווט ובכל מסך יוכל המשתמש לעסוק בפעילות אחרת (משמאל – שרטוט ראשוני של הרעיון):
1) דף בית – מסך עם המבחנים הקרובים שהוכנסו למערכת. לחיצה על מבחן לוקחת ל"מסך הלמידה".
2) דף החיפוש – ניתן לחפש בקלות קבצים (סיכומים, מצגות, תמונות) שאנשים העלו ואפילו חומרים שונים שכבר קיימים באפליקציה ומאושרים על ידי צוות הפיתוח.
3) דף שאלונים – בדף זה ניתן למצוא שאלונים תחרותיים שונים בנושאים שונים. ניתן לענות על כל שאלון רק פעם אחת ולנסות לשבור את השיא הקיים (נמדד הזמן שלוקח לסיים את השאלון).
4) דף העלאה חדשה – הדף בו מעלים סיכום או מצגת חדשה מהטלפון.
5) דף הפרופיל שלי – השם שלי וקבצים שהעלתי. מאוד נוח לעקוב אחר הקבצים שלי ולהוריד אותם לכל מכשיר בכל מקום בעולם.

ב. פירוט היכולות

כאמור, האפליקציה תאפשר מספר פיצ'רים ותכונות שונות למשתמש. הנה רשימה שלהם. יתכן ובגרסה הסופית רשימה זו תשתנה.
- תלמיד יוכל להכניס למערכת את המבחנים הקרובים שלו ולקבל סיכום מדויק של החומר עליו המבחן, בנוסף לשאלות על החומר.
- המערכת תציע לתלמיד ללמוד למבחן או להנות משאלון קצר עליו.
- תלמיד יוכל לענות על שאלון אמריקאי מהיר כהכנה לפני מבחן – שאלון שייבנה אוטומטית ויותאם אישית לחומר עליו המבחן של התלמיד.
- כל משתמש יוכל לשתף חומר, סיכומים ומצגות באופן קל ונוח עם כל המשתמשים או להעלות קובץ באופן פרטי (רק הוא רואה אותו).
- כל משתמש יוכל לחפש סיכומים שאנשים אחרים העלו בנושא מסוים ואף להוריד אותו לטלפון.
- כל משתמש יוכל לענות על שאלון אמריקאי תחרותי על נושא כללי ולנסות לעשות זאת בזמן הקצר ביותר. שלושת המהירים ביותר יופיעו ברשימה לפני שאנשים אחרים עונים על השאלון.

רשימת יכולות מפורטת יותר ניתן לראות בפרק הבא – פרק ג' – מסמך הניתוח.

ג. פירוט הבדיקות

בטבלה הבאה ניתן יהיה לראות בדיקות שונות שאני מציע שיש לערוך על המערכת שלי על מנת לוודא כי עובדת. כמובן, זוהי רשימה חלקית בלבד וכדי לבדוק שהאפליקציה שלי עובדת, יש לנסות ולבצע את כל התכונות שהיא מבטיחה לעשות – ולראות אם היא מצליחה.

מספר	שם הבדיקה	מה אמורה לעשות בדיוק	איך מתכננים לבדוק
1.	הוספת משתמש	לוודא כי הוספת משתמש עובדת כמצופה	יש לנסות ולהירשם כמשתמש חדש, להתנתק ולהיכנס מחדש עם אותו המשתמש.
2.	הוספת מבחן	לוודא כי ניתן להוסיף מבחן לרשימת המבחנים	יש לנסות להוסיף מבחן בתור משתמש מסוים. לוודא כי נמצא ברשימת המבחנים. להתנתק ולהיכנס שוב – לוודא כי הוא שם!
3.	למידה למבחן	לוודא כי דף הלמידה עובד כראוי	לאחר יצירת מבחן, יש לוודא כי הגישה לדף הלמידה עובדת כראוי. לוודא כי מוצגות שם שאלות, אפשרות לענות על שאלון אמריקאי, הסיכום וכדומה.
4.	העלאת סיכום	לוודא כי ניתן להעלות סיכום בהצלחה	לוודא כי מסך העלאת הקבצים עובד כראוי ואכן מוסיף את הקובץ למסד הנתונים. לראות כי הקובץ אכן קיים בפרופיל המשתמש.
5.	הורדת סיכום	לוודא כי ניתן להוריד סיכום בהצלחה.	מציאת סיכום באפליקציה (שאחרים העלו או שהאפליקציה מכינה אוטומטית). לוודא כי ניתן להוריד אותה באופן מוצלח למקום הנבחר בטלפון מתוך האפליקציה.
6.	לקיחת שאלון אמריקאי	לוודא כי השאלון האמריקאי עובד כראוי.	מתוך מסך הלמידה, לחיצה על כפתור ספציפי אמור לקחת אותנו למשחק קטן של שאלון אמריקאי. לוודא כי השאלון נוצר משאלות רלוונטיות לחומר של המבחן עליו השאלון וכי השאלון רץ ועובד טוב ובאופן חלק.
7.	חיפוש	לוודא כי אופציית החיפוש אכן עובדת כמצופה	לחפש תכנים מסוימים ולראות אם התוצאות אכן תואמות למצופה – מוצגות כל התוצאות ורק התוצאות הרלוונטיות.

ד. תכנון לוח זמנים לפרויקט

פעילות	זמן התחלה מתוכנן	זמן סיום מתוכנן	זמן התחלה בפועל	זמן סיום בפועל	הערות
1. בחירת רעיון	נובמבר 2018	נובמבר 2018	-	-	BCademy
2. חקר ראשוני	01.12.2018	15.12.2018	29.11.2018	12.12.2018	
3. פיתוח גרסת MVP ראשונית (צד שרת ואפליקציה)	15.12.2018	15.01.2019	01.01.2019	08.01.2019	
4. חקר מעמיק – Flutter, Django	16.01.2019	31.01.2019	09.01.2019	29.01.2019 (ותוך כדי הפיתוח)	
5. הרחבת השרת וסיום	01.02.2019	15.02.2019	01.02.2019	14.02.2019	נמשכה העבודה גם אחרי
6. הרחבת האפליקציה	16.02.2019	31.03.2019	16.02.2019	09.04.2019	עבודה גם על השרת
7. טיפול סופי בבאגים ותקלות	חודש אפריל	חודש אפריל	10.04.2019	הגשת הפרויקט (תמיד יש מה לשפר)	

ה. ניהול סיכונים בפרויקט

הסיכון	פירוט הסיכון	רמת הסיכון	דרכי התמודדות	מה בוצע בפועל	תאריך
Flutter	זוהי פלטפורמה חדשה לגמרי. כמעט ואין מידע באינטרנט.	קשה	1. מעבר על הדוקיומנטציה של גוגל על Flutter. 2. לוודא כי אני מבין את פלאטר טוב מאוד לפני תכנות האפליקציה (טוטוריאלים וכדומה)	1. יצירת אפליקציה בסיסית (דוגמא של גוגל) לצורך דוגמא. 2. למידה תוך כדי מהדוקיומנטציה של גוגל.	לאורך הפיתוח
זמן	האם יהיה מספיק זמן לתכנות כל מה שארצה?	בינוני	1. עבודה מהירה, יעילה, ללא התברברות או עצלנות. לא לוותר! 2. אם נתקעים – לעבוד על משהו אחר ולחזור בשלב מאוחר יותר. 3. לשאול בפורומים, חברים ומנטור מפייסבוק אם נתקעתי במשהו.	1. תכנון זמנים יסודי ועמידה בלוח הזמנים שלי. 2. לא התברברתי.	לאורך הפיתוח.
גודל האפליקציה	האפליקציה גדולה מאוד, מסועפת ודורשת תכנון מוקפד	בינוני	1. תכנון מוקפד של האפליקציה, המראה שלה והפונקציונליות שלה, כולל התקשורת עם השרת, לפני תחילת התכנות. 2. עזרה במנטור מפייסבוק. 3. עזרה מחברים ומפורומים באינטרנט (Stack Overflow...)	1. תכנון מוקפד של האפליקציה. 2. שינוי מבנה האפליקציה תוך כדי הפיתוח מלווה בתיעוד מתאים (בשרטוטים שלי).	עם תחילת הפיתוח – תחילת חודש ינואר 2019.
שרת גדול	השרת, גם הוא, גדול מאוד, מסועף ומכיל הרבה מידע	קל	1. תכנון מוקדם מוקפד. לחשוב על כל המקרים ועל התקשורת עם צד הלקוח. 2. עבודה בסביבה נוחה שמאפשרת שינויים מהירים (Django). 3. לא למהר ולקחת את הזמן בתכנון ובבדיקות, תוך כדי הכתיבה. 4. שימוש בשרת שיצליח להחזיק הרבה מידע ויהיה נגיש (לקנות שרת?)	1. ווידאתי שאני מבין איך לעבוד עם Django לפני תחילת העבודה. 2. בניתי שרת טוב, שניתן לשנות ולהוסיף לו תכונות ויכולות תוך כדי העבודה.	עם תחילת הפיתוח ותוך כדי הפיתוח.

פרק ג' – BCademy – מסמך ניתוח

בחלק זה של תיק הפרויקט, נעסוק באופן נרחב יותר ביכולות המערכת ונדבר על התהליכים המרכזיים והיכולות המרכזיות שיכולה לבצע המערכת. נציג באופן מובהר וברור את יכולות המערכת, ועבור כל יכולת נתאר את מהותה, את התהליכים שמבצעת ואף מבחינה תכנותית – באילו אובייקטים ורכיבים עושה שימוש. ניתן לראות דיאגרמת Use Case בסוף הפרק. דיאגרמה של כל המחלקות השונות ניתן לראות בסוף פרק ד'.

<u>הרשמה למערכת ויצירת משתמש חדש</u>
היכולת הבסיסית ביותר – יצירת משתמש חדש וכניסה למערכת עם משתמש קיים. כמובן, עושה שימוש באובייקטים של משתמש (User) בצד השרת כדי לנהל את כל המשתמשים. כך, לכל משתמש חווייה ייחודית משל עצמו – מבחנים משלו, קבצים משלו וכו'.

<u>למידה למבחן</u>
היכולת הבסיסית ביותר של האפליקציה. מאפשרת לתלמיד לבחור מבחן מתוך מסך הבית (מבחן שיצר בעצמו – ראה "יצירת מבחן") ולקרוא את החומר הרלוונטי. יוצגו לתלמיד כרטיסיות והתלמיד יוכל לעבור ביניהן – כך, במקום לקרוא את כל החומר כמקשה אחת, החומר יוצג לתלמיד בשלבים, כרטיסייה אחת אחרי השנייה. בנוסף, ניתן לשלוח את כל החומר לאימייל מסוים, לגישה נוחה או להדפסת סיכום.

יכולת זו מתקשרת עם השרת ומבקשת את כל רשימת תתי הנושאים ופירוט החומר הרלוונטי למבחן מסוים – עושה זאת על ידי בקשת GET רגילה לקישור מסוים. השרת מחזיר, בתור מילון שעבר סוריאליזציה של Json, את כל החומר הרלוונטי. לאחר מכן, החומר מוצג בצורה נוחה למשתמש, כמתואר.

כפי שניתן להבין, בקוד עצמו (גם של האפליקציה וגם של השרת) ישנו אובייקט של מבחן (Test), כאשר בין היתר יש לו תכונה של "סיכום" – רשימת כל תתי הנושאים (SmallTopic) שקשורים אליו (בקשר של Many To Many בצד השרת – ב-Database).

<u>מענה על שאלון אמריקאי לקראת מבחן</u>
אחת היכולות המעניינות והכיפיות ביותר באפליקציה – מעבר מהיר על שאלון אמריקאי שמותאם אישית למבחן אותו בחר התלמיד. האפליקציה מתקשרת עם השרת ומבקשת את כל השאלות הרלוונטיות למבחן מסוים ולכל שאלה 4 תשובות – אחת נכונה ו-3 שגויות. המשתמש יכול להינות מהמשחק, לענות על השאלות וללמוד למבחן. בסוף השאלון יוצגו למשתמש על כמה שאלות הצליח לענות בניחוש ראשון וכמה ניחושים שגויים ביצע.

כמובן, גם כאן יש צורך בשימוש באובייקט של מבחן. לכל מבחן גם קשורים נושאים (SmallTopics) ולכל נושא שכזה קשורים שאלות (לכל שאלה יש נושא אחד שמתאים לה). כך, השרת יכול, בבקשת האפליקציה (שליחת בקשת GET לקישור מסוים שחלק ממנו זה ה-ID של המבחן), לספק רשימה של כל השאלות הרלוונטיות למבחן מסוים (כמובן, בשרת נעבוד עם אובייקט של Question – עוד עמודה בטבלת ממסד הנתונים שלנו. לכל אובייקט כזה יש את השאלה עצמה, את הנושא שאליה קשורה, 3 תשובות שגויות ותשובה נכונה). האפליקציה מקבלת מהשרת את השאלות והתשובות (שוב, בתור מילון שעבר סוריאליזציה של Json), מציגה את השאלות במשמש צבעוני וכיף ומאפשרת להנות ממענה על השאלון.

יצירת מבחן חדש

אחת היכולות הבסיסיות של האפליקציה – מאפשרת לבחור מקצוע, נושאים ספציפיים במקצוע ותאריך. לאחר בחירת כל הגורמים הרלוונטים, יוצרת את המבחן ומוסיפה אותו לדף הבית (שם ניתן לראות את כל המבחנים הקרובים).

יכולת זו מתקשרת עם השרת ומעורב בה אובייקט מרכזי אחד – המבחן (גם בצד השרת וגם בצד האפליקציה). בעת יצירת המבחן, המערכת מורידה מהשרת את רשימת הנושאים (SmallTopic) המתאימה למקצוע (Subject) הנבחר ומתוכם בוחר המשתמש אילו נושאים רלוונטים למבחן. יצירת המבחן עצמו נעשית על ידי בקשת POST ובה מילון (שוב, Json) הכולל את כל המידע הרלוונטי ליצירת מבחן. כמובן, השרת מחזיר תשובה אם יצירת המבחן צלח או לא.

העלאת קובץ לשרת

יכולת מעניינת של האפליקציה – להעלות קבצים שישמרו על הענן. כל משתמש יכול לבחור מאין הוא רוצה להעלות את הקובץ (מיקום על הטלפון), לאיזה מקצוע הקובץ קשור (אזרחות, ספרות...), לבחור אם יהיה פומבי או פרטי ולתת על הקובץ מידע נוסף.

כמובן, יש אובייקט של מסמך (Document) במסד הנתונים שלנו – בצד השרת. לאובייקט כזה יש את ה-url המלא לקובץ ששמור מקומית על המחשב/שרת, שם הקובץ ותאריך שבו נוסף למערכת. בנוסף, כמובן, ישנו המידע הנוסף על הקובץ – מידע אותו הוסיף המשתמש (טקסט). בבקשת משתמש – ניתן להעלות קובץ, כל עוד יש חיבור לאינטרנט. העלאת הקובץ נעשית בשליחת בקשת POST מרובת חלקים לשרת, כאשר המידע על הקובץ מחולק בין השליחות השונות. כרגע, הגבלתי את גודל הקובץ המירבי ל-100 MB.

הורדת קובץ

האפליקציה מאפשרת להעלות קבצים לשרת, אך גם להוריד ממנו קבצים מקומית לטלפון! כך, מכל מקום בעולם, כל אחד יוכל להתחבר עם המשתמש שלו לאפליקציה, לצפות ולהוריד את הקבצים הנחוצים לו.

כמובן, גם כאן יש תקשורת עם השרת, והאובייקט הרלוונטי הוא אובייקט הקובץ/מסמך (Document). ישנו מסך שלם שעוסק בקבצים של המשתמש – כל הקבצים מסודרים לפי סדר הוספה. עבור כל קובץ מופיע שמו ומידע נוסף עליו, כפי שהוכנס על ידי המשתמש. ניתן להוריד כל קובץ למכשיר על ידי שליחת בקשת GET מתאימה לשרת לקישור הרלוונטי. השרת יגיב בתשובה מרובת חלקים – כל המידע של הקובץ.

חיפוש קבצים ונושאים

אופציה נוספת העומדת בפני המשתמש היא לחפש קובץ שמשתמש אחר העלה או נושא שקיים באפליקציה. האפליקציה תציג למשתמש תוצאות משני סוגים – חומר רלוונטי שקיים באפליקציה (קבצים) או סיכומים קשורים (בנושא או בתוכן). אופן החיפוש יעשה על ידי הכנסת טקסט חיפוש.

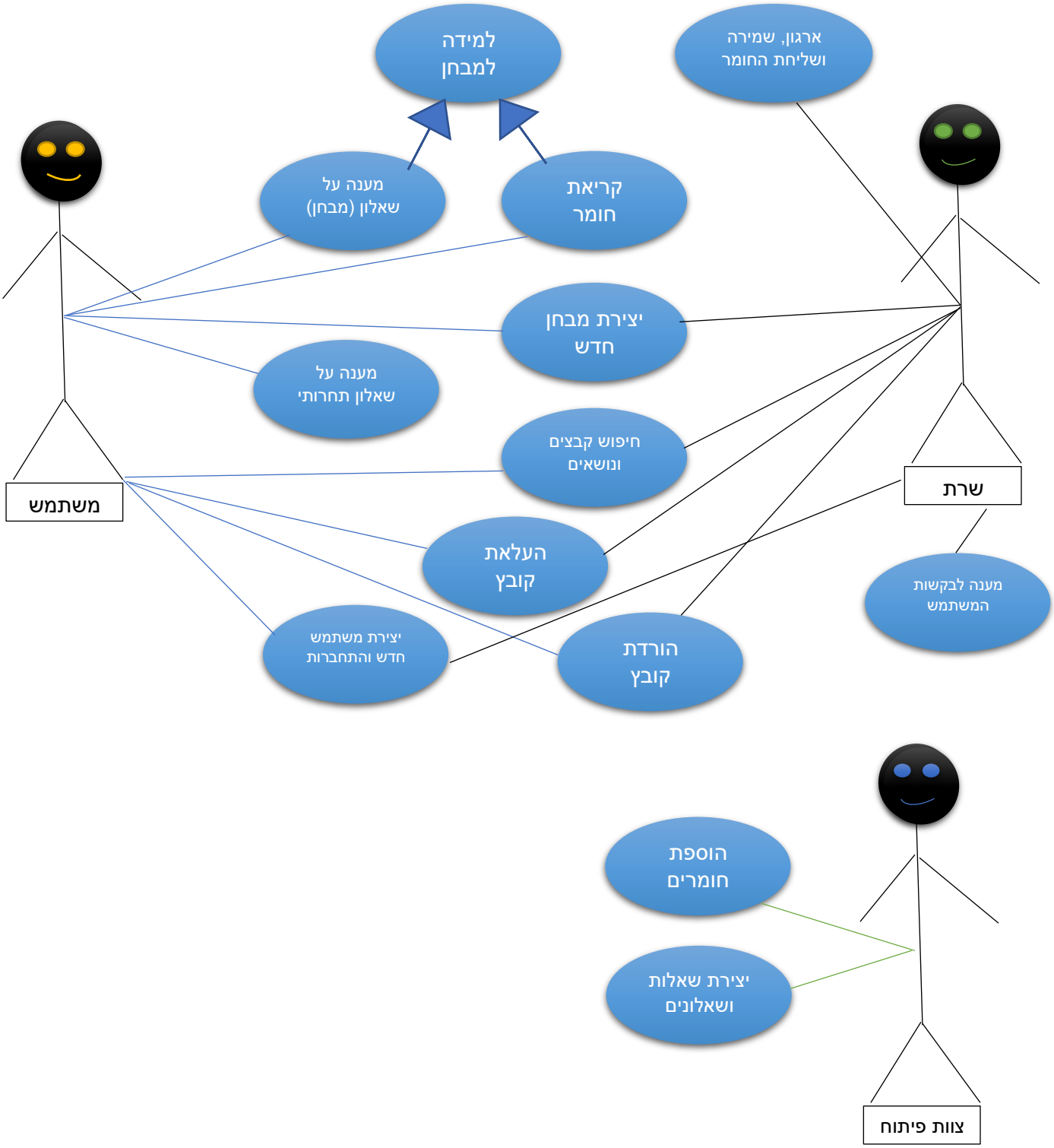
כמובן, זה יבוצע על ידי בקשה לשרת – שליחת בקשת GET לקישור רלוונטי. מעורבים כאן שוב – אובייקטי המסמכים (Document) והנושאים (SmallTopic). השרת יגיב בכל סוג חיפוש בתשובה אחרת, אך בכל מצב ישלח מילון שעבר סוריאליזציה כ-Json.

מענה על שאלון תחרותי

התכונה האחרונה שהוכנסה לאפליקציה ומוסיפה לשיתופיות – אפשרות לענות על שאלונים אמריקאים ולהתחרות על המקומות הראשונים. כל משתמש יכול לבחור מקטלוג של שאלונים ולענות על שאלון מסוים. לפני המענה על השאלון יראה כמה זמן לקח לשלושת הטובים ביותר וינסה להביס אותם ולענות מהר יותר. בסוף השאלון יוצג למשתמש כמה זמן לקח לו והאם הצליח להיכנס לשלישייה הראשונה ובאיזה מקום.

כאן לוקחים חלק אובייקטים של שאלון (Quiz), השייך למקצוע מסוים (Subject) וכמובן יש לו את השאלות הקשורות אליו (Question). גם כאן, השרת שולח בקשת GET לקישור המתאים ומקבל כתשובה את כל השאלות הקשורות לשאלון ואת שלוש השחקנים הטובים ביותר והתוצאות שלהם.

על מנת לסכם את כל היכולות הללו מבחינת השרת ומבחינת המשתמש, אראה דיאגרמת Use Case קצרה המתארת את כל יכולות המערכת בצורה ברורה. הדיאגרמה תראה מה יכול משתמש לעשות באפליקציה (מה השימושיות שלה) ובאופן כללי מה עושה השרת מאחורי הקלעים. בנוסף, מוצג תפקיד של צוות המפתחים – להיות אחראים על הוספת חומר למאגר ויצירת שאלות ושאלונים.



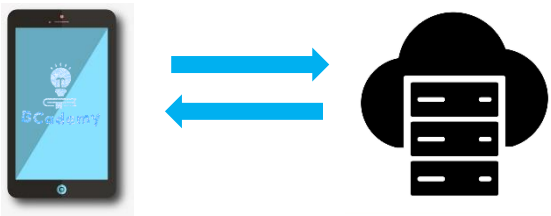
פרק ד' – BCademy – העיצוב

א. תיאור הארכיטקטורה הכללית

המערכת שלי בנויה משני רכיבים עיקריים:
- אפליקציה שנכתבה בשפת Dart (נקראת ככה כי הקוד נראה כמו חץ בצורה שלו) באמצעות פלטפורמת Flutter היחסית חדשה של גוגל (הסבר נוסף בהמשך). ניתן להריץ את האפליקציה על כל טלפון Android או iOS (על מנת להריץ על אייפון נדרש מחשב מסוג Mac).
- צד שרת שנכתב בשפת Python באמצעות פלטפורמת Django. Django מאפשר תחזוקה יחסית קלה של השרת ומונע התעסקות בבניית טבלאות ה-Database בעצמנו. השתמשתי בשרת ש-Django מגיע איתו אשר רץ על המחשב הרצוי (המחשב משמש כשרת) וב-Database של sqlite.

בקוד של צד השרת ובקוד של צד האפליקציה מוגדרים מחלקות ואובייקטים דיי דומים. שני הצדדים מדברים ביניהם באמצעות תקשורת בפרוטוקול HTTP (צד האפליקציה מעביר בקשות POST או GET), ומעבירים ביניהם יחידות מידע העוברות סוריאליזציה באמצעות json בבקשות ובתשובות ה-HTTP.

בהמשך יוצגו המחלקות השונות וכיצד שמור המידע בצד השרת. בנוסף לכך, נוכל לראות כיצד צד האפליקציה מנהל את המידע.



ב. תיאור הטכנולוגיה בשימוש

כאמור, באפליקציה שלי השתמשתי ב-2 טכנולוגיות עיקריות – Flutter לפיתוח האפליקציה, Django לפיתוח השרת.

הבחירה ב-Flutter לפיתוח האפליקציה לא הייתה מובנת מאליה בכלל מבחינתי. בזמן בו בחרתי לעבוד עם Flutter היא עוד הייתה בגרסת הביתא שלה, וכשהתחלתי לעבוד בדיוק שוחררה גרסה 1.0 – הגרסא היציבה הראשונה.

העבודה עם Flutter היוותה אתגר בעצמה מכיוון שכמעט ולא היו תשובות באינטרנט לשאלות שונות. זה כמובן באופן שונה מטכנולוגיות אחרות שקיימות כבר שנים, ועל כל שאלה יש תשובה (או לפחות כמעט כל שאלה). בנוסף, הטכנולוגיה עובדת עם שפת התכנות Dart של גוגל – לכן, הייתי צריך ללמוד את השפה (קצת פחות בעייתי).

ואמנם, בחרתי ב-Flutter כי זו פלטפורמה חדשה ותמיד רציתי להיות חלוץ בטכנולוגיה כלשהי – וזו הייתה ההזדמנות שלי. בנוסף, אני מאמץ אתגרים ואוהב לצאת מאיזור הנוחות – גם אם זה אומר לוותר על עוד פיצ'ר לאפליקציה ולחקור כל הלילה בשביל לפתור באג קטן ומעצבן, אבל לזכות בידע נוסף.

באופן כללי, Flutter זה פריימוורק חדש (גרסא 1.0 יצאה ב-4 בדצמבר 2018) של גוגל לפיתוח אפליקציות לאייפון ולאנדרואיד, על בסיס Open-Source. הפלטפורמה עובדת עם ווינדוס, מאק-או-אס ולינוקס. המנוע כתוב ברובו בשפת c++ ועושה שימוש במנוע הגרפי של גוגל – "Skia". הפלטפורמה מאפשרת פיתוח מהיר וקל של אפליקציות על בסיס ווידג'טים בסיסיים קיימים (עיצובים שקיימים במערכת), או

יצירה של ווידג'טים חדשים על ידי המתכנת. ניתן לעבוד עם פלאטר בכל פלטפורמה וכל סביבת פיתוח, כל עוד הקבצים הדרושים ל-Flutter על מנת לעבוד מותקנים במחשב. כל המידע הרלוונטי על Flutter, כולל הדוקיומנטציה עצמה, נמצא באתר Flutter.dev (ביקרתי באתר הזה הרבה בחודשים האחרונים).

הבחירה ב-Django הייתה יותר קלה מבחינתי מכיוון שכבר הכרתי את הטכנולוגיה והתעסקתי איתה מעט בעבר. כמובן שלא ברמה בה עסקתי בה בחודשים האחרונים, אך כן הכרתי את הטכנולוגיה וידעתי כי היא נוחה ושימושית.

Django, גם הוא מבוסס קוד פתוח, הוא פריימוורק לפיתוח של אתרים ורכיבים העושים שימוש במאגר נתונים אינטרנטי. המטרה העיקרית שלו היא להפוך את היצירה של מאגרי מידע מורכבים לפשוטה יותר ומסייע במיפוי אובייקטים לבסיס נתונים יחסי (ORM). נכון להיום, הגרסא היציבה האחרונה של Django היא 2.2 ואיתה עבדתי. בטכנולוגיה זו כותבים בשפת Python האהובה, והיא מבוססת על פרוטוקול HTTP לתקשורת בין הרכיב הטכנולוגי/אפליקציה/אתר לבין השרת שמנהל את בסיס הנתונים. הפלטפורמה הבסיסית של Django מציעה שרת בסיסי חינמי שנכתב על ידי הצוות של Django המסייע בבחינה וניסוי של הקוד שנכתב, כך שכתוכניתן יכלתי להתעסק רק בכתיבת ניהול בסיס הנתונים ולא להתעסק ביצירת התקשורת.

העבודה עם Django יכולה להיעשות גם היא בכל סביבת פיתוח כל עוד Django מותקן על המחשב. Django תומך בווינדוס, לינוקס ומק-או-אס. האתר שלהם בכתובת www.djangoproject.com (כולל הסברים ודוקיומנטציה מפורטת).

כאמור, רוב המידע שעובר בין השרת לאפליקציה בבקשות ובתשובות ה-HTTP עובר סוריאליזציה ב-Json. Json זה קיצור של JavaScript Object Notation ומשתמשים בו לסוריאליזציה של מידע מסוגים שונים (מילונים, רשימות וכדומה) על מנת שניתן יהיה לשמור מידע בקובץ או לשלוח אותו כטקסט או כמידע, למשל בבקשות ותשובות HTTP או בכל דרך שהיא. (כמובן, לא ניתן לשלוח סתם מילון או רשימה כחלק מקוד... לכן חשוב להעביר את המידע סוריאליזציה)

ג. תיאור סביבת הפיתוח

כאמור, לכתיבת האפליקציה היו שני צדדים.
על מנת לעבוד על צד האפליקציה בצורה נוחה ביותר (כאמור, בשפת Dart בטכנולוגיית Flutter) כתבתי ב-Android Studio – הפלטפורמה העיקרית לכתיבת אפליקציות לאנדרואיד, וכיום היא כמובן תומכת גם ב-Flutter. בנוסף, אותו ה-IDE מציע גם באופן מובנה אימולטורים של מכשירי אנדרואיד על מנת לבדוק ולנסות את האפליקציה במהלך שלב הפיתוח – נוח מאוד (אין צורך לחבר מכשיר אנדרואיד/אייפון למחשב ולהוריד את האפליקציה).

את צד השרת כתבתי ב-Pycharm – ה-IDE הנפוץ ביותר לכתיבה בשפת Python. כמובן, זאת לאחר התקנת Django למחשב.

על מנת לבדוק את השרת – לשלוח לו בקשות מסוימות ולראות את התשובות – השתמשתי בתוסף לדפדפן גוגל כרום בשם "ARC", המאפשר לשלוח בקשות HTTP מסוגים שונים לכתובות שונות ולשנות פרמטרים שונים בבקשה באופן נוח. כמובן, התוסף מראה גם את התשובה באופן מפורט (תשובת ה-HTTP/S עצמה – כטקסט), או באופן מומחש (כפי שניתן לראות בדפדפן).

ד. דיון באלגוריתמים השונים בקוד

האפליקציה אותה פיתחתי עושה שימוש במספר אלגוריתמים שונים המנהלים את האפליקציה – הן בצד השרת והן בצד האפליקציה וממשק המשתמש. רוב האלגוריתמים אינם מסובכים ברמה המתמטית, אלא עוסקים בעיצוב האפליקציה, הצגת מידע רלוונטי למשתמש וניהול המידע המתקבל ו/או נשלח מהאפליקציה.

כך למשל, ישנם אלגוריתמים רבים בצד השרת שאחראים על סידור החומר הרלוונטי למבחן מסוים ועל שליחתו בסדר הנכון לאפליקציה בעת בקשה. בצד האפליקציה, ישנם אלגוריתמים שיודעים לקלוט את החומר הנשלח, לעבד את המידע בצורה הנכונה ולהציג אותו באופן נוח וקל לקריאה. דוגמא נוספת היא היכולת להעלות ולהוריד קבצים, ליצור שאלונים שונים ולהציגם בצורה אינטראקטיבית למשתמש, יצירת מבחנים והצגתם וכדומה. כלומר, רוב האלגוריתמים המסובכים ביותר בקוד עובדים על התקשורת בין האפליקציה והשרת ועל הצגת המידע באופן נוח למשתמש.

אתן דוגמא לבעיה שעלתה לי במהלך הפיתוח הקשורה גם באלגוריתם – כיצד לגרום לחלק הלמידה להיות קצת פחות משעמם? בתחילת הפיתוח, חלק הלמידה באפליקציה היה החלק הפחות כיף באפליקציה, ועכשיו הוא החלק הכיפי ביותר. כיצד עשיתי זאת? ראשית, חילקתי את קטע הלמידה לשני חלקים – שאלון אמריקאי וחלק הלמידה עצמה. בניית השאלון האמריקאי לא היוותה בעיה מיוחדת, אך לא ידעתי כיצד להציג את חלק הלמידה (הטקסט – החומר עצמו) באופן מעניין.

כגרסא ראשונית, כל החומר פשוט הוצג לקורא והוא יכל היה לקרוא אותו כמקשה אחת. ואמנם, רציתי לפתח את האלגוריתם שמציג את החומר ולהפוך את העיצוב למזמין ומהנה יותר לקריאה. לכן, שיניתי את הקוד והפכתי את חלק הלמידה להרבה יותר מזמין מבחינה ויזואלית – כל תת נושא מוצג בכרטיסייה והקורא יכול לדפדף בין כרטיסיות שונות. כך, לא כל החומר מוצג לקורא ב"בום", והוא יכול להיחשף אל החומר בהדרגתיות.

ישנם קטעי קוד רבים הפותרים בעיות שונות, אותם ניתן יהיה לראות בפרק הבא – פרק ה' – הקוד. ביניהם קודים שיודעים לסדר את תוצאות החיפוש שהגיעו מהשרת בסדר הנכון ולהציג אותם בצורה נוחה, או אפילו איזה כפתור להציג למשתמש בהתאם למצבים שונים של האפליקציה. בפרק זה ישנו גם הסבר על כל קטע קוד, כל אלגוריתם, על אופן הפעולה שלו וכיצד הוא פותר בעיה קיימת.

ה. תיאור מודולים וספריות בהן נעשה שימוש

במהלך כתיבת האפליקציה, עיצובה ושיפורה, עשיתי שימוש במספר ספריות שונות שאפשרו לי להוסיף שורות מעטות לקוד ועם זאת, להוסיף תכונות גדולות לאפליקציה. בחלק זה, נדון בהרחבה בכל המודולים וספריות בהן נעשו שימוש, תוך תיאורן, הן בצד השרת והן בצד האפליקציה.

- צד השרת:

-ספריות שונות של Django ובהן:

<u>models:</u> מודול בסיסי של Django שהכרחי ליצירת מחלקות שונות של פייתון המהוות את העמודות הבסיסיות ביותר בטבלת הנתונים שלנו. כל מחלקה בקוד יורשת מ-models.Model והתכונות שלה הן סוגים שונים של models – טקסט (models.CharField), מספר (models.IntegerField) וכדומה. כך, Django יודע שאותן מחלקות הן חלק מבסיס הנתונים שלנו.

<u>get_object_or_404</u>: פונקציה המנסה לאתר אובייקט ממחלקה מסוימת בבסיס הנתונים על פי קריטריונים מסוימים המועברים לה כפרמטרים (לדוגמא – מספר id, תכונה מסוימת של האוביקט כמו שם וכדומה). הפונקציה מחזירה את האובייקט אם מוצאת אותו. במקרה ולא מוצאת את האובייקט, מוחזרת תשובת HTTP 404 – NOT FOUND.

HttpResponse, HttpResponseBadRequest, HttpResponseServerError: מאפשרות להחזיר תשובות HTTP מסוגים שונים (200, 400, 500). לוקחות כפרמטר את המידע הרצוי (טקסט או מידע שעבר סוריאליזציה) בתשובת ה-HTTP.
timezone: מודול המאפשר לעסוק בקוד בתאריכים, שעות וקטעי זמן. שימושי בקוד שלי, למשל, כאשר יוצרים מבחן חדש ומעניקים לו תאריך. דוגמא נוספת היא כאשר משתמש מבקש לראות את כל המבחנים שלו – רק מבחנים שעתידים להגיע (החל מהיום הנוכחי והלאה) יוצגו.
ObjectDoesNotExist: מאפשר לאתר שגיאה בה אובייקט מסוים אותו חיפשנו לא נמצא בבסיס הנתונים ולהחזיר תשובה בהתאם.

-ספריות שאינן קשורות ל-Django:

json: מאפשר לקודד ולפענח קטעי טקסט באמצעות ג'ייסון המוכר והאהוב.
os: עוסקת בקשר בין תהליכים שונים. מאפשרת לקוד לבצע פעולות שונות במערכת ההפעלה (למחוק, ליצור ולשנות מיקום של קבצים, תיקיות וכדומה). כאשר נוצר/נמחק קובץ באפליקציה – יש שימוש במודול זה בשרת.
mimetypes: מאפשר לקבל את ה-mime type המתאים לתשובת HTTP כאשר משתמש רוצה להוריד קובץ לטלפון. שימושי כאשר משתמש מבקש להוריד קובץ מסוים – בתשובת ה-HTTP ישנו סוג הקובץ.
datetime: מאפשר לעסוק בתאריכים וזמנים באופן שאינו קשור לאובייקטים בבסיס הנתונים.

- צד האפליקציה (ספריות של flutter ו-dart):

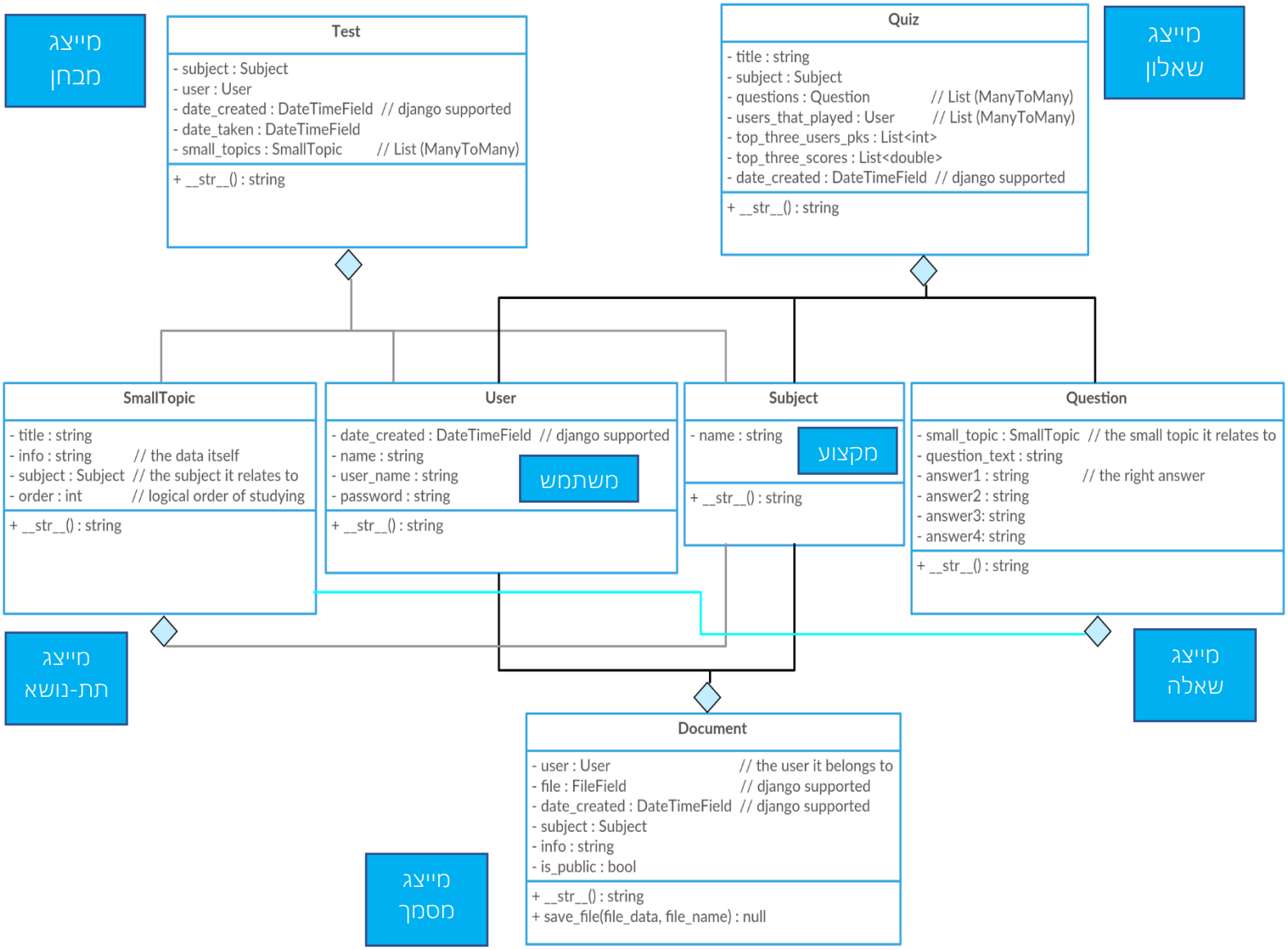
material: ה-import הבסיסי בכל דף. מאפשר לנו להשתמש באבני הבנייה הבסיסיים ביותר לפיתוח ב-Flutter – בווידג'טים.
async: מאפשר למספר פונקציות לרוץ במקביל (דוגמא: להוריד קובץ תוך כדי שהמשתמש עדיין יכול להשתמש באפליקציה). שימושי בייחוד כאשר המשתמש מבצע פנייה לשרת, ואנו רוצים שעדיין ניתן יהיה להשתמש באפליקציה כאשר שליחת הבקשה, קבלת התשובה ועיבודה מתבצעים ברקע.
cupertino: מאפשר להשתמש בווידג'טים (אבני עיצוב) הנראים כאילו נלקחו ממערכת ההפעלה iOS.
auto_size_text: מאפשר לטקסט לשנות באופן אוטומטי את גודל הפונט ואת מספר השורות על מנת לא לחרוג מהמקום שהוקצה לו באפליקציה. שימושי במקרים בהם אורך הטקסט יכול להשתנות כתוצאה מקלט של המשתמש.
io: ספרייה העוסקת בקשרים בין תהליכים ובין האפליקציה לבין הטלפון עצמו (שמירה, מחיקה ויצירה של קבצים, תיקיות וכדומה. דומה ל-os בפייתון). שימושי ביצירה ומחיקה של קבצים מהטלפון.
http: כפי ששמה רומז – מאפשרת לעסוק בצורה קלה ונוחה בבקשות וקבלת תשובות HTTP.
path_provider: מספקת את ה-path לתיקיות שימושיות שונות במערכת המכשיר (תיקייה לשמירת קבצים, תיקייה של תמונות שמורות וכדומה).
flutter_calendar_carousel: ספרייה המאפשרת להציג לוח שנה על המסך ובאמצעותו לבחור תאריך. מאפשרת שינוי העיצוב של לוח השנה. נעשה בספרייה שימוש בחלק של יצירת מבחן, שם המשתמש בוחר תאריך מתאים למבחן.

open_file: כשמו, מאפשר לפתוח קובץ (לאחר שסופק המיקום שלו על המכשיר) ולצפות בו.
iterables: ספרייה העוסקת במספר של מבני נתונים (מילונים, רשימות וכו'). מאפשרת, בין היתר, לעבור בלולאת for אחת על שתי רשימות בו זמנית (פונקציית zip).
services: מאפשר לדעת אם שירות/יכולת מסוימת קיימים במכשיר (ולהימנע מלנסות להשתמש בשירות אם לא קיים).
file_picker: מאפשר לפתוח file explorer של הטלפון ולבחור קובץ מסוים. המיקום של הקובץ מסופק לאפליקציה כ-return של פונקציה. שימושי כאשר המשתמש בוחר קובץ אותו רוצה להעלות לענן.
flutter_mailer: ספרייה המספקת מחלקה מסוג אימייל. התכונות של המחלקה הן: רשימת נמענים, תוכן, נושא, כתובות קבצים מצורף (אם יש) ורשימת נמעני עותק (למי לשלוח עותק). בנוסף, מספקת פונקציה ששולחת את האימייל הזה (בשם מפתיע – send). כלומר, זו ספרייה שמאפשרת לשלוח אימייל היישר מהקוד, כל עוד יש חיבור אינטרנט פעיל.

ו. תיאור מבני הנתונים

בתרשים ה-Class Diagram UML הבא, ניתן לראות באופן כללי את כל המחלקות הקיימות בקוד שלי. ניתן לראות את הקשרים השונים ביניהם ובאופן כללי כיצד הם מתקשרים אחד עם השני. כמובן, תרשים זה עוזר מאוד להבין כיצד המערכת מבצעת את מה שהיא מבצעת, וגם להבין את הפרק הקודם שעוסק ביכולותיה של המערכת.

חשוב להבהיר שהמחלקות שניתן לראות כאן כתובות בצד השרת (Django מאפשר להתייחס לבסיס הנתונים ולאובייקטים בו כאל אובייקטים של מחלקות – נוח מאוד!) ונעשה בהם שימוש גם בצד האפליקציה – לא בהכרח באופן "מחלקתי" (אין צורך מבחינה עיצובית).



* כאשר כתוב "Django supported" הכוונה במודול של Django שנעשה בו שימוש.
** הכוונה ב-ManyToMany הוא בקשר בין האובייקטים בבסיס הנתונים.

פרק ה' – Bcademy – הקוד

בפרק זה נעבור יחד על 8 קטעי קוד שונים מהפרויקט שלי, נסביר בקצרה על היכולת שהם מבצעים ועל הייחוד שלהם בעיניי ו/או מבחינת הפרויקט. בפרק זה ננסה לשים את האצבע על מה שהופך את הפרויקט שלי למיוחד – מבחינת הקוד ומבחינת הבנייה שלו. צילומי מסך של האפליקציה המתאימים לקטעי הקוד ניתן יהיה לראות בפרק ז' – מדריך למשתמש.

חלק א' – קטעי קוד מהשרת

1. יצירת משתמש חדש

בקטע הקוד הזה, שהוא פונקציה אחת מתוך פונקציות רבות שנכתבו בתוך השרת שלי, ניתן לראות כיצד אני יוצר משתמש חדש – תכונה מאוד בסיסית של האפליקציה (ובגלל זה חשובה).

```
def create_user(request):
    """
    Creates a new user with the info. Checks whether the username is taken.
    :param request: the HTTP get request
    :return: http response.
    """
    if request.method == 'POST':  # make sure it's a post.
        new_user = 0
        try:
            info = json.loads(request.body)

            # Check if username is already taken.
            if User.objects.filter(user_name=info['user_name']).count():
                return HttpResponseBadRequest(json.dumps(
                    {"error": "username taken"}))
            else:
                # Create the new user
                new_user = User(name=info['name'],
                                user_name=info['user_name'],
                                password=info['password'])
                new_user.save()
                return HttpResponse(json.dumps({
                    'name': new_user.name,
                    'pk': new_user.pk
                }))
        except KeyError as e:  # In case we didn't get the info as expected
            if isinstance(new_user, User):
                new_user.delete()  # Delete the new user we created
            return HttpResponseBadRequest(
                "Could not create user - data was not good: " + str(e))
    else:  # In case we got a GET request.
        return HttpResponseBadRequest(
            "Should be a POST request (got something else).")
```

הפונקציה מקבלת כפרמטר את request – בקשת ה-HTTP כפי שהתקבלה בשרת. בגוף בקשת ה-HTTP ישנם כל הנתונים על המשתמש החדש (שם מלא, שם משתמש, סיסמא). הפונקציה קוראת את הנתונים יוצרת משתמש חדש. הפונקציה מחזירה תשובה (מילון שעובר סוריאליזציה ב-Json) ובה השם של המשתמש וה-pk (מספר מזהה – ID) שלו. כך, האפליקציה יכולה לשמור את הנתונים הללו עליו ולשייך כל תשובה למשתמש מסוים.

כמובן, הפונקציה בודקת שכל המידע הדרוש ליצירת משתמש חדש התקבל, שבקשת ה-HTTP היא מסוג POST וששם המשתמש לא תפוס. במידה וישנה שגיאה מסוימת – מחזירה תשובת HTTP מתאימה שמפרטת את השגיאה.

2. חיפוש נושאים

בקטע הקוד הבא ניתן לראות כיצד נעשה ביצוע חיפוש הנושאים בשרת – חלק מהיכולות של האפליקציה. המשתמש מכניס מילת חיפוש ומקבל את כל הנושאים התואמים לו.

גם כאן, מקבלת הפונקציה כפרמטר את request (מוסבר עליו בקטע הקוד הקודם) ובגוף בקשת ה-POST את מילת החיפוש.

נערך חיפוש נושאים שבהם הכותרת מכילה את מילת החיפוש ושהנושא שלהם מתאים למילת החיפוש. היא מחזירה מילון ובו ערכי המפתחות הם pk- (מספר סידורי, id) של כל נושא והערך הוא רשימה שמכילה את הכותרת והמקצוע המתאים לנושא (כך מוצגת הרשימה באפליקציה).

כמובן, הפונקציה לוקחת בחשבון מקרי קצה ושגיאות שונות (לא נמצאו תוצאות, המידע שנשלח לשרת לא כמצופה, נשלחה בקשת GET ולא POST). השרת מחזיר תשובת HTTP מסוג מתאים ומפרט את הבעיה/שגיאה.

```
def search_small_topics(request):
    """
    Returns pks and info about small topics related to the search word.
    :param request: the HTTP request.
    :return: HTTP response
    """
    if request.method == 'POST':
        try:
            search = json.loads(request.body)['search']   # The search word
            if search:
                # search for subject
                subject_related = SmallTopic.objects.filter(
                    subject__name__contains=search)
                # search in title
                topics_related = SmallTopic.objects.filter(
                    title__contains=search).order_by('subject')
                final_result = {}
                # get the results
                for result in subject_related:
                    final_result[result.pk] = [result.title,
                                               result.subject.name]
                for result in topics_related:
                    final_result[result.pk] = [result.title,
                                               result.subject.name]
                return HttpResponse(json.dumps(final_result,
                                               ensure_ascii=False))
            else:  # If no search word
                return HttpResponseBadRequest("Must search for something.")
        except KeyError as e:
            return HttpResponseBadRequest(
                "Did not get the search word. " + str(e))
    else:
        return HttpResponseBadRequest(
            "Should be a POST request (got something else).")
```

3. העלאת קובץ

קטע הקוד הבא מעלה את המידע של קובץ ושולח אותו לאפליקציה, כך שהמשתמש יוכל להוריד קובץ היישר מהשרת ומכל מקום בעולם.

השרת מקבל כחלק מהקישור את ה-pk של הקובץ הרצוי ומחזיר כתשובה תשובת HTTP עם המידע של הקובץ ופרמטרים הכוללים גם את סוג הקובץ, כך שהקובץ יוכל להישמר במכשיר עם הסיומת הנכונה שלו (קובץ וורד, PDF וכדומה).

במקרה של שגיאה (התקבלה בקשת POST במקום GET, הקובץ לא אותר בשרת) מוחזרת תשובת HTTP מתאימה למשתמש עם פירוט השגיאה.

בשל בלבול של Django עם השגת המיקום של קובץ, הוחלף בקוד התו ':' ברצף התוים "%3A" (אסקי), ועל כן ביצעתי את החילוף שוב בעצמי.

```
def download_file(request, doc_pk):
    """
    Uploading a file for the user to download.
    :param request: the HTTP request
    :param doc_pk: the pk of the document the user wants to download
    :return: HTTP response
    """
    if request.method == 'GET':
        # Getting the document object and the path to it
        doc = get_object_or_404(Document, pk=doc_pk)
        doc_path = doc.file.url.replace("%3A", ":") \
            if "%3A" in doc.file.url else doc.file.url

        if os.path.isfile(doc_path):
            # Creating the right response with the file data
            with open(doc_path, 'rb') as f:
                # The data and the mimetype
                response = HttpResponse(
                    f.read(),
                    content_type=mimetypes.guess_type(doc_path)[0]
                )
                # Make sure it's a file and add its name
                response['Content-Disposition'] = \
                    'inline; filename=' + os.path.basename(doc_path)
                return response
        else:
            return HttpResponseServerError(
                "Could not locate file. See: " + doc_path)
    else:
        return HttpResponseBadRequest(
            "Should be a GET request (got something else).")
```

חלק ב' – קטעי קוד מהאפליקציה

4. ה-main

קטע קוד לא מסובך בכלל אך קריטי לקיום האפליקציה – קטע הקוד שרץ עם הרצת האפליקציה.

בקטע קוד זה אנו יוצרים את הבסיס לאפליקציה, מחליטים מה יהיה דף הבית, מה יהיה הרקע שלה, איך יהיה העיצוב הבסיסי שלה, מה שמה וכו'. בנוסף, אנו מגדירים דפים נוספים באפליקציה אליהם נרצה לקפוץ ממקום אחר במהלך הריצה.

```
/// The function that's called when we run the app
void main() => runApp(BCademy());


class BCademy extends StatelessWidget {
  // The root of the application.
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      title: 'BCademy',
      theme: ThemeData(
        backgroundColor: Color(0xffffebee),
        primarySwatch: Colors.blue,
        fontFamily: 'Rubik',
      ), // ThemeData
      home: SignInPage(), // the starting page
      // routes (pages) we use in the app
      routes: <String, WidgetBuilder> {
        '/tests': (BuildContext context) => new HomePage(),
        '/quizzes': (BuildContext context) => new HomePage(startPage: 2,),
      },
    ); // MaterialApp
  }
}
```

5. הורדת קובץ

חלק מה-API של האפליקציה שמאפשר תקשורת עם השרת והורדה של קובץ. הפונקציה מקבלת את ה-pk (id) של הקובץ ושם שבו תרצה לשמור את הקובץ.

לאחר פנייה לכתובת המתאימה בשרת, השרת מחזיר תשובה ובה המידע על הקובץ. המידע יורד מהשרת, נפתח קובץ חדש והמידע נכתב אליו. הפונקציה מחזירה (במקרה שהכל עובד) את המיקום בו שמור הקובץ כדי שהמשתמש יוכל לפתוח אותו. אם קרה משהו לא תקין, היא מחזירה טקסט (סטרינג) ריק ומדפיסה הודעת שגיאה מתאימה (במקרה שהסטטוס קוד של תשובת ה-HTTP אינו 200, או במקרה שישנה כל שגיאה אחרת ביצירה ושמירת הקובץ).

```
/// Download a file to the phone
Future<String> downloadFile(filePk, fileName) async {
  try {
    // Get all file info (bytes) from the server
    final url = Uri.http(_url, '/bcademy/download/$filePk/');
    var response = await http.get(url);

    // Make sure we got an OK
    if (response.statusCode == 200) {
      var bytes = response.bodyBytes;  // file data
      String dir = (await getExternalStorageDirectory()).path + '/Download';
      File file = new File('$dir/$fileName');
      await file.writeAsBytes(bytes);  // write to the file
      return '$dir/$fileName';         // return the path to the new file
    } else {
      print("Something went wrong getting the file $fileName.");
      return '';
    }
  } catch (e) {  // in case of error
    print("Something went wrong with saving the file $fileName. " +
        e.toString());
    return '';
  }
}
```

6. סיכום חומר למבחן

קטע קוד זה מראה מה קורה מאחורי הקלעים כאשר משתמש מבקש לראות סיכום לקראת מבחן עתידי.

לאחר שהמידע הנחוץ יורד מהשרת, ישנה פונקציה שאחראית ליצור כרטיסים, כאשר כל תת נושא מוצג בכרטיסייה משלו (כך, במקום לראות את החומר במקשה אחת, ניתן לדפדף בין כרטיסיות). הכרטיסיות שמורות ברשימה במשתנה infoCrads_.

כאשר המשתמש מבקש לראות את הסיכום, מוצגת לו הכרטיסייה המתאימה במרכז המסך (ברירת המחדל היא הכרטיסייה הראשונה, באינדקס 0), מספר הכרטיסייה בו נמצא מתוך סך הכרטיסיות (בחלק המסך העליון), ושני חצים – חץ ימינה שמאפשר לעבור לכרטיסייה הבאה בלחיצה עליו וחץ שמאלה שמאפשר לעבור לכרטיסייה הקודמת בלחיצה עליו.

במקרה של בעיה, כאשר אין מידע (רשימת הכרטיסיות ריקה, או שהאינדקס של הכרטיסייה בה אנו נמצאים מחוץ לגבולות הרשימה – לא אמור לקרות) הפונקציה תציג הודעה למשתמש שמשהו שגוי קרה, כך שיוכל לנסות לטעון את האפליקציה או את העמוד בשנית.

```
/// Present a card with some of the material. The user can go to the
/// next/previous cards by pressing arrow buttons.
Widget _getSummary() {
  // Make sure we have some topics (should be)
  if (_infoCards.isNotEmpty && _index >= 0 && _index < _infoCards.length) {
    // return all the widgets organized
    return Stack(
        children: <Widget>[
          // tells us what number of card we're watching
          // out of the total number of cards
          Align(
              alignment: Alignment(0, -0.95), // location on screen (x, y)
              child: Text("${_index+1}/${_infoCards.length}",
                style: TextStyle(fontSize: 16.0),) // Text
          ), // Align
          // the card with the material itself
          Align(
            alignment: Alignment(0, 0),  // location on screen (x, y)
            child: SingleChildScrollView(
              child: Center(
                child: Padding(
                  padding: const EdgeInsets.only(top: 40.0, bottom: 10.0),
                  child: Padding(
                      padding: EdgeInsets.fromLTRB(12.0, 0, 12, 12),
                      child:_infoCards[_index] // the correct card
                  ), // Padding
                ) // Padding
              ), // Center
            ), // SingleChildScrollView
          ), // Align
          // a right arrow sign. press to move to the next card (if possible)
          Align(
            alignment: Alignment(1.05, 0),
            child: IconButton(
              icon: Icon(Icons.arrow_forward_ios, color: Colors.grey[800],),
              onPressed: () {
                setState(() {
                  // increase number of card we're looking at if possible
                  if (_index < _infoCards.length - 1)
                    _index++;
                });
              },
            ), // IconButton
          ), // Align
          // a left arrow sign. press to move to the previous card
          // (if possible)
          Align(
            alignment: Alignment(-1.05, 0),
            child: IconButton(
              icon: Icon(Icons.arrow_back_ios, color: Colors.grey[800],),
              onPressed: () {
```

```
              onPressed: () {
                setState(() {
                  // decrease number of card we're looking at if possible
                  if (_index > 0)
                    _index--;
                });
              },
            ), // IconButton
          ), // Align
        ] // <Widget>[]
      ); // Stack
  }
  // in case there are no topics, return text that tells the user that
  // something went wrong (there are small topics in each test).
  else {  // should never come here
    return Center(child: Container(child: Text("Something went wrong...")),);
  }
}
```

7. תוצאות חיפוש של קבצים

קטע הקוד הבא מציג לנו מה קורה מאחורי הקלעים כאשר משתמש מחפש קבצים מסוימים באפליקציה ורואה רשימה של אותם הקבצים. _search זו מילת החיפוש של המשתמש ו-_files זה תוצאת החיפוש.

לפני שחיפש משהו – מציג למשתמש טקסט שהוא מוזמן לחפש...
אחרי החיפוש – אם יש תוצאות מציג אותן בגריד של 2 טורים. ניתן לנווט לעמוד בו מוצג מידע על הקובץ ואף להוריד את הקובץ בלחיצה עליו. אם אין תוצאות, מציג הודעה מתאימה שלא נמצאו קבצים מתאימים. במקרה של שגיאה, מציג הודעה מתאימה. (קטע קוד אחר דואג ליידע את המשתמש אם נפלה שגיאה בתקשורת).

```
/// Get a grid of all the files suitable to the search word
Widget _getFileResultsPage() {
  // In case the user did not search for anything
  if (_search == '' || _files == null) {
    // Text that tells the user to search
    return Expanded(
      child: ListView(
          physics: const NeverScrollableScrollPhysics(),
          primary: false,
          shrinkWrap: true,
          children: [
            SizedBox(height: 200.0,),
            Center(child:
            Text("...נסה לחפש משהו",
              textDirection: TextDirection.rtl,
              textAlign: TextAlign.center,
              style: TextStyle(fontSize: 28.0),),), // Text, Center
          ]
      ), // ListView
    ); // Expanded
  }
  // In case there's nothing suitable for the user's search word
  else if (_files.isEmpty) {
    // Tell the user we could not find anything :(
    return Expanded(
      child: ListView(
          physics: const NeverScrollableScrollPhysics(),
          primary: false,
          shrinkWrap: true,
          children: [
            Container(height: 200.0,),
            Center(child:
            Text("לא מצאנו קבצים מתאימים\n😞",
              textDirection: TextDirection.rtl,
              textAlign: TextAlign.center,
              style: TextStyle(fontSize: 28.0),),), // Text, Center
          ]
      ), // ListView
    ); // Expanded
  }
  // In case we found some results - show them to the user
  else {
    try {
      // Make the list of results (will be used in the grid view):
      final fileTiles = <Widget>[];
      for (String pk in _files.keys) {
        Map file = _files[pk];
        fileTiles.add(
            Padding(
              padding: const EdgeInsets.all(8.0),
              child: InkWell(
                // This function will be called when pressed on a file tile:
                onTap: () {
```

```
                _goToFilePage(file);
                },
                child: Container(
                  decoration: BoxDecoration(
                      borderRadius: BorderRadius.circular(8.0),
                      color: Color(0xff80deea)), // BoxDecoration
                  child: Center(child: Column(
                    mainAxisAlignment: MainAxisAlignment.center,
                    textDirection: TextDirection.rtl,
                    // Some info about the file:
                    children: <Widget>[
                      AutoSizeText("${file['subject_name']}",
                        textDirection: TextDirection.rtl,
                        style: TextStyle(fontSize: 30.0),
                        textAlign: TextAlign.center,), // AutoSizeText
                      SizedBox(height: 5.0,),
                      AutoSizeText(file['info'],
                        textDirection: TextDirection.rtl,
                        style: TextStyle(fontSize: 20.0),), // AutoSizeText
                      SizedBox(height: 5.0,),
                      AutoSizeText("${file['name']}",
                        style: TextStyle(fontSize: 14.0),
                        textAlign: TextAlign.center,), // AutoSizeText
                      SizedBox(height: 8.0,),
                      AutoSizeText("${file['day']}."
                          "${file['month']}.${file['year']}",
                        style: TextStyle(fontSize: 14.0),), // AutoSizeText
                    ], // <Widget>[]
                  )), // Column, Center
                ), // Container
              ), // InkWell
            ) // Padding
        );
      }
      // Return the grid view
      return Expanded(
        child: Directionality(
          textDirection: TextDirection.rtl,
          child: GridView.count(
            primary: true,
            shrinkWrap: true,
            crossAxisSpacing: 0,
            crossAxisCount: 2,
            children: fileTiles,
          ), // GridView.count
        ), // Directionality
      ); // Expanded
    }
    catch (e) {
        return Expanded(
          child: ListView(
              physics: const NeverScrollableScrollPhysics(),
              primary: false,
              shrinkWrap: true,
              children: [
                SizedBox(height: 200.0,),
                Center(child:
                Text("): היתה תקלה בקריאת המידע על הקבצים",
                  textDirection: TextDirection.rtl,
                  textAlign: TextAlign.center,
                  style: TextStyle(fontSize: 28.0),),), // Text, Center
              ]
          ), // ListView
        ); // Expanded
    }
  }
}
```

8. קבלת כפתור להורדת קובץ

כאמור, משתמש יכול להוריד קובץ (באמצעות לחיצה על כפתור בעמוד מסוים). כאן, ניתן לראות כיצד התוכנה מחליטה איזה כפתור להציג לנו.

במקרה שעוד לא נלחץ – מציגה כפתור רגיל, המאפשר להוריד את הקובץ בלחיצה עליו (המשתנה שמחליט איזה פונקציה תתרחש בלחיצה על הכפתור, func, משתנה בהתאם).

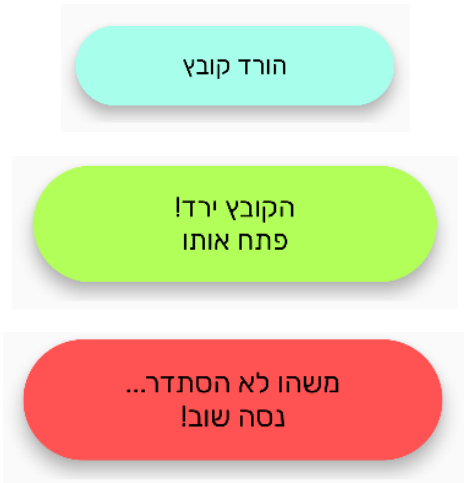
```
/// Get a the button for downloading
Widget _getButton() {
  double height = 75.0;             // height of button
  double width = 300.0;             // width of button
  Color color = Color(0xffa7ffeb);  // color of button
  String text = "הורד קובץ";        // text of button
  var func;                         // the func that'll execute on press

  // In case the download went successfully (after first press)
  if (_afterDownload && _filePath != '') {
    height = 100.0;
    width = 350.0;
    color = Colors.lightGreenAccent;
    text = "הקובץ ירד!\nפתח אותו";
    func = _openFile;
  }
  // In case download went wrong
  else if (_afterDownload && _filePath == '') {
    height = 100.0;
    width = 350.0;
    color = Colors.redAccent;
    text = "משהו לא הסתדר...\nנסה שוב!";
    func = _downloadFile;
  }
  // In case we did not press the button yet
  else {
    func = _downloadFile;
  }

  // The button widget itself:
  return Padding(
    padding: const EdgeInsets.all(16.0),
    child: Material(
        elevation: 10.0,
        borderRadius: BorderRadius.circular(50.0),
        color: Colors.transparent,
        child: AnimatedContainer(
          duration: Duration(milliseconds: 100),
          height: height,
          width: width,
          decoration: BoxDecoration(
              color: color,
              borderRadius: BorderRadius.circular(50.0)
          ), // BoxDecoration
          child: InkWell(
            borderRadius: BorderRadius.circular(50.0),
            onTap: func,  // on tap execute the function
            child: Center(
              child: Text(text,
                textDirection: TextDirection.rtl,
                textAlign: TextAlign.center,
                style: TextStyle(
                    color: Colors.black,
                    fontSize: 24.0
                ), // TextStyle
              ), // Text
            ), // Center
          ), // InkWell
        ) // AnimatedContainer
    ), // Material
  ); // Padding
}
```

במקרה שהכפתור נלחץ וההורדה צלחה – הכפתור משנה את גודלו וצבעו לירוק ומאפשר למשתמש לפתוח את הקובץ בלחיצה עליו.

במקרה של תקלה (ראינו את הפונקציה של הורדת קובץ [מספר 5 בפרק זה] – מחזירה סטרינג ריק במקרה של תקלה בהורדה) – משתנה גודלו של הכפתור, צבעו משתנה לאדום, ומוצגת הודעה על תקלה. הכפתור מאפשר לנסות להוריד את הקובץ שוב.



פרק ו' – BCademy – בדיקות

בפרק זה נחזור לפרק האפיון (פרק ב') בו ראינו טבלה גדולה ובה כל הבדיקות שתכננתי לבצע על האפליקציה על מנת לוודא כי היא עובדת באופן תקין וראויה להפצה ולשימוש נרחב. נסתכל על הטבלה בשנית, ונראה מה בוצע בפועל.

חשוב להדגיש כי הבדיקות הללו נכתבו לפני התחלת העבודה והפיתוח, ומכאן עצם החשיבות שלהן – הן נכתבו לפני שידעתי מה החולשות האפשריות של האפליקציה ולכן הן "נטרליות", ומטרתן לוודא כי האפליקציה אכן עובדת כראוי.

	שם הבדיקה	מטרת הבדיקה	מה נדרש לבצע	מתי	מה בוצע בפועל
1.	הוספת משתמש	לוודא כי הוספת משתמש עובדת כמצופה	יש לנסות ולהירשם כמשתמש חדש, להתנתק ולהיכנס מחדש עם אותו המשתמש.	מיד לאחר פיתוח הפיצ'ר, תוך כדי הפיתוח הכללי ובסיום הפיתוח הכללי	נרשמתי כמשתמש חדש. התנתקתי והתחברתי שוב כאותו משתמש. חזרתי על פעולה זו מספר פעמים.
2.	הוספת מבחן	לוודא כי ניתן להוסיף מבחן לרשימת המבחנים	יש לנסות להוסיף מבחן בתור משתמש מסוים. לוודא כי נמצא ברשימת המבחנים. להתנתק ולהיכנס שוב – לוודא כי הוא עדיין שם!		הוספתי מבחן כמשתמש מסוים. לאחר התנתקות והתחברות מחדש – המבחן עדיין שם. כך עבור מספר משתמשים.
3.	למידה למבחן	לוודא כי דף הלמידה עובד כראוי	לאחר יצירת מבחן, יש לוודא כי הגישה לדף הלמידה עובדת כראוי. לוודא כי יש אפשרות לענות על שאלון אמריקאי ולצפות בסיכום.		עבור כל מבחן שיצרתי עבור כל משתמש – בדקתי שדף הלמידה אכן עובד.
4.	העלאת סיכום	לוודא כי ניתן להעלות סיכום בהצלחה	לוודא כי מסך העלאת הקבצים עובד כראוי ואכן מוסיף את הקובץ למסד הנתונים. לראות כי הקובץ אכן קיים בפרופיל המשתמש.		דרך מספר משתמשים ומספר קבצים מסוגים שונים – העלתי קבצים לשרת. ראיתי כי הם נמצאים בשרת וכי לכל משתמש יש את רשימת הקבצים שלו.
5.	הורדת סיכום	לוודא כי ניתן להוריד סיכום בהצלחה.	מציאת סיכום (קבצים) באפליקציה. לוודא כי ניתן להוריד אותה באופן מוצלח למקום הנבחר בטלפון מתוך האפליקציה.		ווידאתי כי האופציה להוריד קבצים אכן עובדת וכי ניתן לגשת לקבצים שהורדנו דרך הטלפון.
6.	מענה על שאלון אמריקאי	לוודא כי השאלון האמריקאי עובד כראוי.	ניתן לענות על שאלון לקראת מבחן ספציפי, או על שאלון כללי מרשימת שאלונים. לוודא כי השאלון נוצר משאלות רלוונטיות לחומר של המבחן עליו השאלון (אם זה שאלון לקראת מבחן) וכי השאלון רץ ועובד טוב ובאופן חלק.		יצרתי מספר מבחנים ו-ווידאתי כי האופציה לענות על שאלון עובדת טוב. בנוסף, עניתי על מספר שאלונים תחרותיים כמספר משתמשים שונים.
7.	חיפוש	לוודא כי אופציית החיפוש אכן עובדת כמצופה	לחפש תכנים מסוימים ולראות אם התוצאות אכן תואמות למצופה – מוצגות כל התוצאות ורק התוצאות הרלוונטיות.		חיפשתי נושאים וקבצים וראיתי כי מוצגות לי כל התשובות הרלוונטיות לחיפוש שלי.

פרק ז' – BCademy – מדריך למשתמש

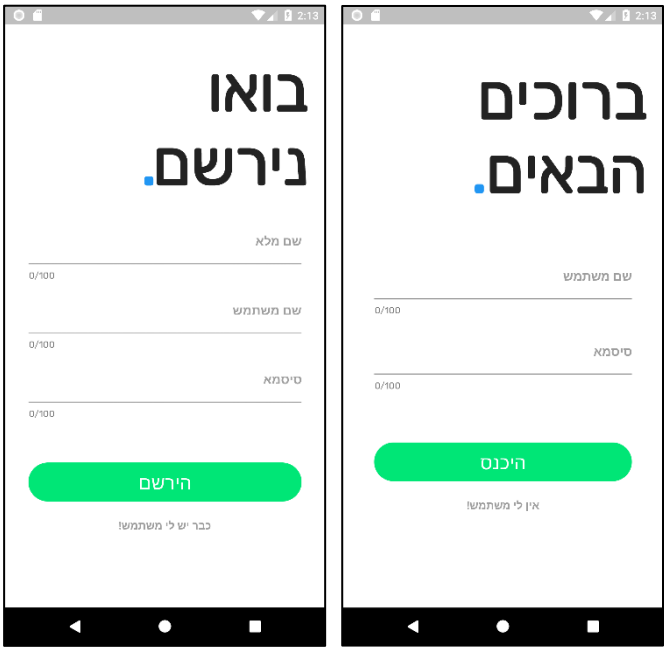
כמשתמשים חדשים, הינכם מוזמנים לצלול אל תוך המדריך המפורט הבא לשימוש באפליקציה החדשה שלי – BCademy. האפליקציה מעוצבת על פי קונבנציות ידועות ודי נוחה לשימוש, אך למתקשים שבינינו – תמיד יש את המדריך למשתמש!

בואו נצא לדרך!

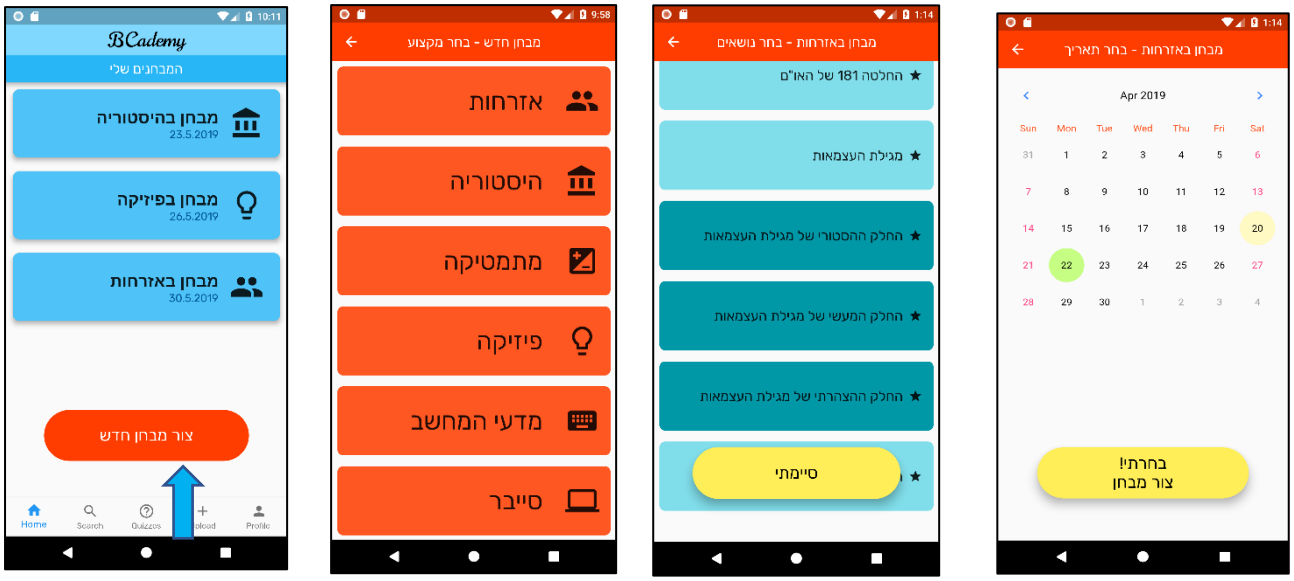
1. הרשמה/התחברות לאפליקציה

על מנת להירשם לאפליקציה בכניסה הראשונה וליצור משתמש חדש, מלאו את כל הפרטים הנחוצים במסך ההרשמה. קלי קלות!

על מנת להתחבר עם משתמש קיים, הכניסו את שם המשתמש שלכם ואת הסיסמא שבחרתם לעצמכם בעת יצירת המשתמש.



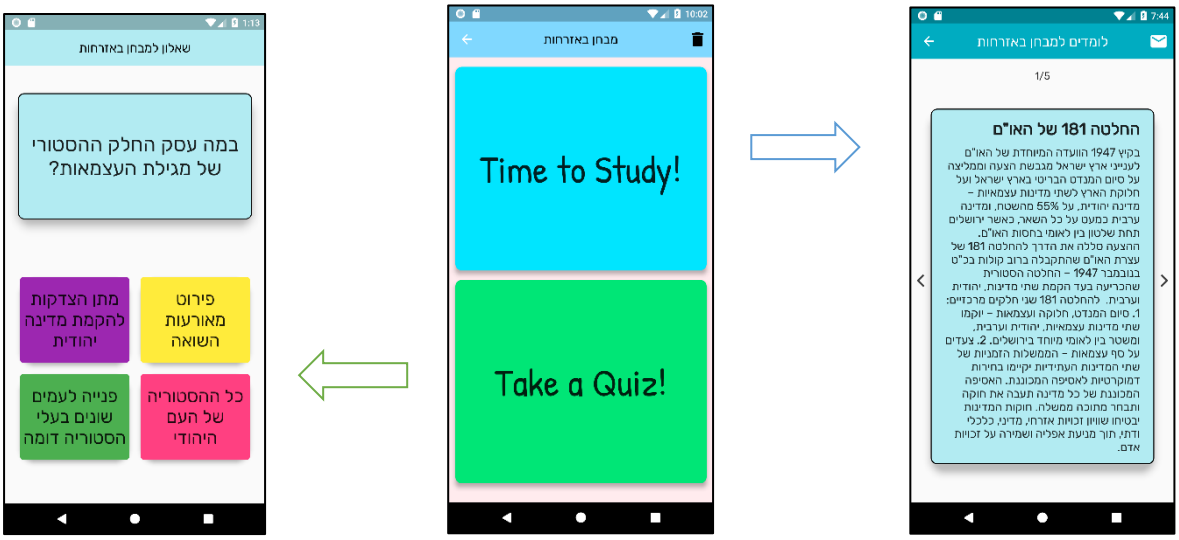
1. יצירת מבחן חדש



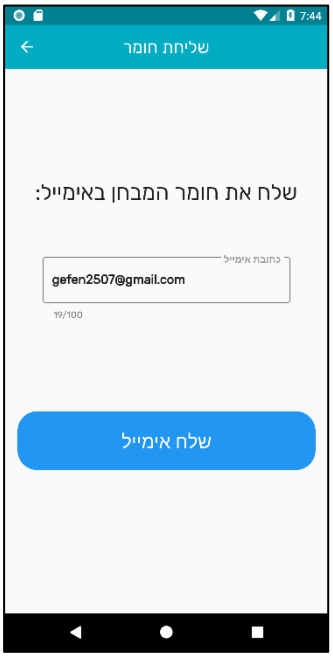
במסך הבית תוכלו לראות רשימה של כל המבחנים שלכם. על מנת ליצור מבחן חדש, לחצו במסך הבית על הכפתור "צור מבחן חדש", בחרו במקצוע הנבחר, בחרו מתוך רשימת הנושאים את הנושאים למבחן , בחרו תאריך ו... סיימתם! המבחן החדש יופיע במסך הבית יחד עם רשימת שאר המבחנים שלכם.

2. למידה למבחן ומענה על שאלון אמריקאי

לחצו על המבחן הנחוץ מתוך רשימת המבחנים ותגיעו למסך ניווט. לחצו על "Time to Study" ותגיעו למסך מיוחד בו תראו את כל החומר למבחן בכרטיסיות ותוכלו לדפדף קדימה ואחורה ביניהן! שם, תוכלו ללמוד לקראת המבחן שלכם. לחצו על "Take a Quiz" ותגיעו לשאלון ששואל אתכם בדיוק על החומר למבחן ושאלות שבאמת יעזרו לכם!



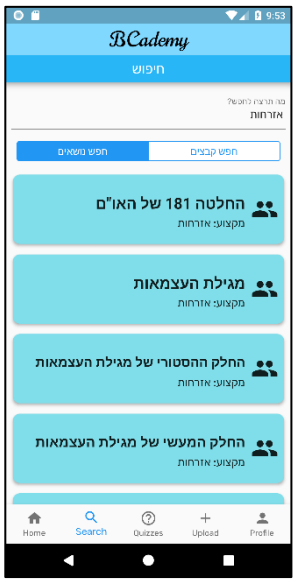
בנוסף, ניתן לשלוח לעצמכם או לאחד החברים את כל החומר באמצעות אימייל. איך? לחצו על האייקון בפינה הימנית העליונה במסך הלמידה, מלאו בדף שנפתח את כתובת האימייל הרצויה ולחצו "שלח אימייל". לאחר מכן, יפתח מסך דרכו תוכלו לצפות באימייל ולשלוח אותו.

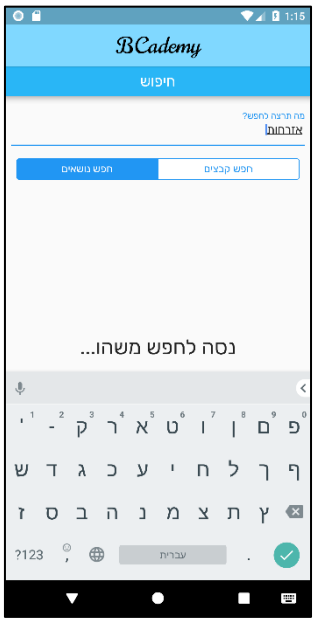


3. חיפוש נושאים וקבצים

כן, כן... ניתן גם לחפש נושאים למבחן וגם לחפש קבצים מסוימים שאתם חושבים שיכולים לעזור לכם (ממש כמו בגוגל!). רק הכניסו את מילת החיפוש (מקצוע מסוים, שם של נושא לימוד וכדומה), בחרו את אופציית החיפוש הרצויה (לחפש קבצים או לחפש נושאים?) וקבלו את התוצאות!

לחיצה על נושא מהתשובות תיקח אתכם לדף בו תוכלו לקרוא עליו. לחיצה על קובץ מרשימת התוצאות תיקח אתכם לדף בו תוכלו למצוא עוד מידע על הקובץ ואפילו להוריד אותו ולצפות בו. פשוט יותר מזה אין....

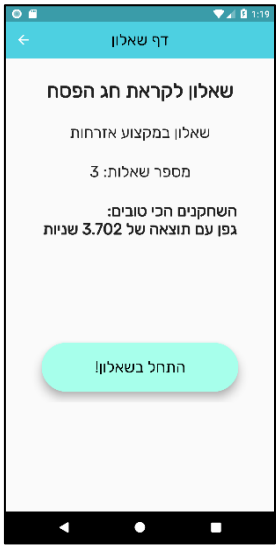


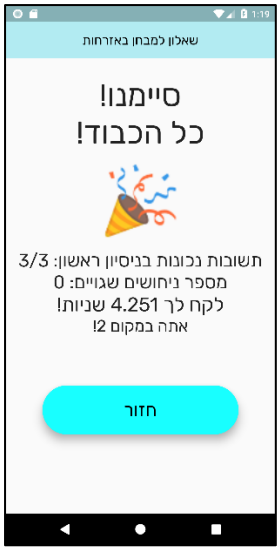


4. מענה על שאלון תחרותי

קודם ראינו כי ניתן לענות על שאלון המתאים לי בדיוק למבחן.... אבל מה אם סתם בא לי לענות על שאלונים וללמוד דברים חדשים? בדיוק בשביל זה ישנו הדף הזה – מענה על שאלונים תחרותיים! כאן, תוכלו לגלול ולחפש ברשימה של שאלונים שונים (שמתעדכנים כל הזמן), להסתכל על כל שאלון ולענות עליו.

אבל שימו לב! כאן ניתן לענות על השאלון רק פעם אחת, נמדד הזמן שלוקח לכם להגיע לסוף השאלון ובסופו של דבר אתם תדורגו לפי התוצאה שלכם! הרי לא סתם קוראים לזה "שאלון תחרותי...".



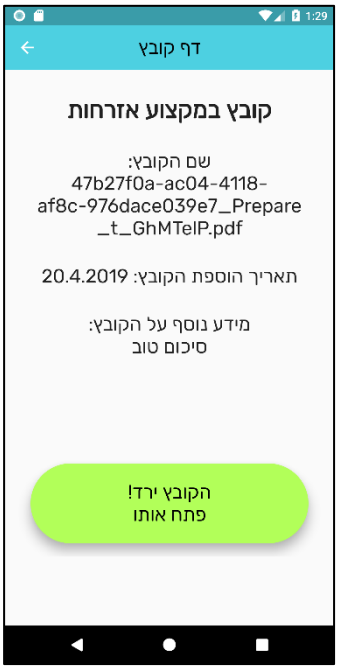


5. העלאת קובץ

בנוסף לכל, תוכלו להעלות קבצים משלכם לענן ולצפות בהם מכל מקום שתרצו! כדי להעלות קובץ בחרו אותו, מלאו את כל המידע הנחוץ, בחרו אם תרצו שיהיה פומבי (כולם יוכלו לחפש ולמצוא אותו) והעלו אותו. כזה פשוט!

6. צפייה בקבצים שלי והורדת קובץ

כמובן, בעמוד הפרופיל שלכם תוכלו לראות את כל הקבצים שהעלתם, לצפות במידע עליהם וכמובן, להוריד אותם לטלפון שלכם – ישירות מהענן. כך, תוכלו להתחבר למשתמש שלכם באפליקציה מכל מכשיר בעולם ולגשת לקבצים שלכם מכל מקום!



פרק ח' – BCademy – מבט אישי

פיתוח האפליקציה היה מאתגר מאוד עבורי – אך מהנה במיוחד. מאז ומתמיד רציתי לדעת לבנות אפליקציות ובאמת לבנות אחת – וזו פעם ראשונה שבה אני באמת בונה אפליקציה שימושית, מסועפת ומפותחת. השקעתי שעות רבות, מחשבה רבה והקרבתי הרבה על מנת לפתח את האפליקציה כך שכל דבר הכי קטן ימצא חן בעיניי. לכן, אני גאה בעבודתי ומאמין כי התוצאה טובה במיוחד.

כמובן, במהלך הפיתוח נתקלתי בקשיים ואתגרים – מלבד הבאגים הקטנים והמעצבנים, הייתי צריך ללמוד טכנולוגיות חדשות לגמרי. את Flutter, כולל שפת התכנות Dart לא הכרתי בכלל! כך, מלבד הכתיבה של האפליקציה עצמה, הייתי צריך להתמודד גם עם הלמידה של הטכנולוגיה והסתגלות לשפה החדשה... בנוסף, הייתי צריך להעמיק קצת יותר ב-Django על מנת להצליח לבנות את כל מה שרציתי.

את כל אלה הייתי צריך להספיק בזמן יחסית קצר – ארבעה חודשים פחות או יותר. למידה של Dart, Flutter ו-Django, פיתוח האפליקציה, פיתוח השרת וכתיבת תיק הפרויקט. כל זאת בנוסף לבדיקות שונות של השרת והאפליקציה.

ואמנם, למרות הלחץ בזמן (יש הרי עוד דברים להספיק ב-יב'...) – נהניתי מאוד בכתיבת האפליקציה והפרויקט. כמו שכתבתי קודם לכן – אני אוהב אתגרים, אוהב להוציא את עצמי מאיזור הנוחות ולהתנסות בדברים חדשים. הפרויקט וכתיבתו היו מעין רכבת הרים בשבילי – היו רגעים של עליות והצלחות, תקופות בהן היה לי זמן רב לעבוד והכל רץ חלק. ואמנם, היו גם רגעי שפל, כמו בזמן לחוץ במיוחד או תקופת מבחנים, בהם כמעט ולא היה לי זמן לעבוד. חוסר הזמן והרצון שלי לעבוד על הפרויקט התנגשו אחד בשני פעמים רבות, ולרוב הפרויקט ניצח.

בנוסף, אני מוכרח להוסיף מספר מילים על המנטור שלי מחברת פייסבוק – רועי יודסין. במהלך החודשים האחרונים הוא ליווה אותי ב-2 פגישות שונות במשרדים של פייסבוק וסייע לי להתגבר על בעיות שונות שצצו במהלך הפיתוח. המפגשים סייעו לי והנחו אותי לעבר תהליך עבודה נכון בכתיבת פרויקטים גדולים.

אני בטוח שבהמשך אני אמשיך לפתח את האפליקציה – אוסיף פיצ'רים נוספים (צ'אט מובנה, רשימות תפוצה של קבצים, שליחת הודעות וכדומה). בנוסף, אני מעוניין להפיץ את האפליקציה ולהעלות אותה לאפ-סטור ולפליי-סטור – אני באמת מאמין כי האפליקציה אכן יכולה לסייע לרבים להתכונן וללמוד למבחנים, וכן להצליח בהם בצורה טובה, יעילה וכיפית יותר. בנוסף, הגשתי פנייה למשרד החינוך וייתכן ויום אחד אפתח אפליקציה רשמית עבורם שתסייע לרבים ללמוד למבחנים ולהגיע לתוצאות טובות יותר.

לסיכום, אני מאמין כי האפליקציה והשרת שכתבתי הם ללא פשרות בשום מקום. עד הרגע האחרון שיפצתי, הוספתי, שיניתי והסרתי חלקים שלא אהבתי או אהבתי פחות. הרעיון לאפליקציה היה רעיון שחשבתי עליו כבר משנה שעברה, ואני גאה בעצמי על כך שהבאתי את הרעיון למציאות. אין דבר כיף יותר מלהיות בסוף של הפיתוח, כאשר המוצר עובד ועובד טוב, והנה – אנחנו כאן! כמובן, יש עוד עבודה ותמיד יש מה לשפר, אך אני חושב שאקח לי הפסקה קצרה אחרי השנה העמוסה הזו (:.

תודה רבה על הקריאה!

פרק ט' – BCademy – ביבליוגרפיה

לקראת הפיתוח למדתי ממספר אתרים שונים.

Dart ו-Flutter:

1. למידת השפה של Dart
https://www.dartlang.org/guides/language/language-tour

2. טוטוריאל קצר של גוגל (בסיסי מאוד) על Flutter
http://classroom.udacity.com/courses/ud905

3. כל הווידג'טים הבסיסיים ביותר של Flutter
https://flutter.dev/docs/development/ui/widgets

4. כל הדוקיומנטציה של Flutter

https://flutter.dev/

5. מידע על ספריות ותוספים של Dart ושל Flutter

https://pub.dev

Django:

6. טוטוריאל בסיסי על Django ודוקיומנטציה (אתר רשמי של Django)

https://www.djangoproject.com/start/

7. סוגים וטיפוסים של מאפיינים שיכולים להיות לאובייקטים של Django

https://docs.djangoproject.com/en/2.1/ref/models/fields/

כמובן, בנוסף לכל היו עשרות ואולי אפילו מאות חיפושים שונים בגוגל וביקורים באתרים שונים, שם עונים על שאלות משתמשים או מפרסמים כתבות על טכנולוגיות שונות:

8. https://medium.com/ - כתבות ובלוגים (בנושאים שונים, גם טכנולוגיה)

9. https://stackoverflow.com/ - תשובות לשאלות משתמשים בנושא תכנות וטכנולוגיה

בנוסף, נעזרתי מעט במנטור מפייסבוק ובחברים.

הסוף.