**Deployment**

**הקדמה:** האתר פועל באופן מלא תחת שירותי הענן של Google Cloud.

Project name: KidsRead

Project ID: kidsread

Project number: 35788023348

**Google Cloud App Engine**

* ביצוע Deployment של האתר ל – App Engine
* יש להתקין Google Cloud SDK
* נכנסים לאפליקציה שהותקנה ומנווטים לתיקיית BookReading
* מעניקים הרשאות גישה בעזרת הפקודה :

gcloud auth login

עבור החשבון kidsread.appspot@gmail.com(סיסמא מופיעה בקובץ app.standard.yaml תחת המשתנה GMAILPW)

* מריצים את הפקודה gcloud app deploy app.standard.yaml
* בסיום התהליך השינויים יעלו לענן
* ניתן להכנס לאתר בכתובת : [https://kidsread.appspot.com](https://kidsread.appspot.com/)
* מידע נוסף ניתן למצוא ב <https://cloud.google.com/appengine/docs/standard/nodejs/building-app/deploying-web-service>
* ניתן לראות logs של האתר בעזרת הפקודה gcloud app logs tail -s default באפליקציית

Google Cloud SDK.

**Google Cloud SQL**

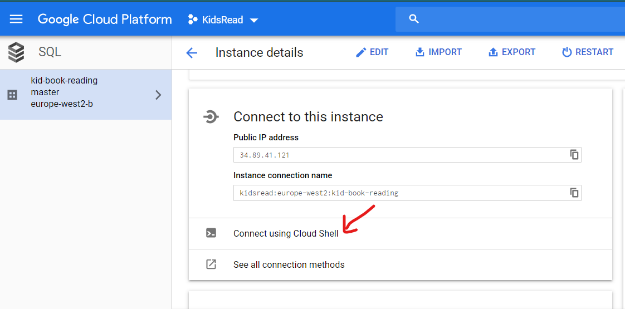
קיים instance של PostgreSQL בענן:

Instance id: [kid-book-reading-db](https://console.cloud.google.com/sql/instances/kid-book-reading-db/overview?project=kidsread&authuser=1)

Location: europe-west2-b

Public IP address: 34.89.41.121

Instance connection name: kidsread:europe-west2:kid-book-reading

* כיצד להתחבר ל DB באמצעות Cloud Shell:
* נכנסים Google Cloud SQL ולוחצים :
* ­­­מתחברים לשרת באמצעות הפקודה :

gcloud sql connect kid-book-reading --user=postgres --quiet

מכניסים סיסמא עבור המשתמש postgres (מופיע בקובץ app.standard.yaml תחת המשתנה SQL\_PASSWORD)

מתחברים לבסיס הנתונים באמצעות הפקודה:

\c kidBookReadingDB postgres

מכניסים סיסמא עבור המשתמש postgres (מופיע בקובץ app.standard.yaml תחת המשתנה SQL\_PASSWORD)

כעת נוצר חיבור לבסיס הנתונים וניתן להריץ פקודות של PostgreSQL.

* מבנה ה DDL שאיתו נבנה בסיס הנתונים נמצא בתיקיית docs : kidBookReadingDDL.sql

על מנת לבנות בסיס נתונים חדש ניתן להשתמש בכל הפקודות שבקובץ זה. (יש לשים לב שאם היה מידע אמיתי של משתמשי האתר אזי מידע זה ימחק)

* אתחול ראשוני שהוכנס לבסיס הנתונים נמצא בתיקיית docs : initializeDB.sql

ניתן להשתמש בקובץ זה כדי להכניס מידע ראשוני לבסיס הנתונים. (יש לשים לב שאם היה מידע אמיתי של משתמשי האתר אזי מידע זה ימחק)

* על מנת לגבות את הבסיס הנתונים שקיים בענן ניתן להעזר במדריך זה :  
  <https://cloud.google.com/sql/docs/postgres/import-export/exporting>

**Google Cloud Storage**

קיים bucket בשם kidsread.

קבצים שמועלים באופן דינמי באתר מועלים ל bucket זה בעזרת הפונקציה : **uploadToCloud**  אשר נמצאת ב: src\queries.js.

מידע נוסף על אופן העלאת הקבצים ניתן למצוא כאן: <https://cloud.google.com/appengine/docs/standard/nodejs/using-cloud-storage>

**פיתוח ותחזוקה**

סביבת הפיתוח בצד השרת היא Node.js. שרת ה web שנבחר הינו express.

ה-View engine של הדפים הדינמיים הוא hbs, וכל הדפים הדינמיים בעלי סיומת זו (.hbs) נמצאים בתיקיית templates.

על מנת לשמור על סדר ביצענו חלוקה לקבצים נפרדים עבור כל קבצי js וקבצי ה css לרוב הדפים הדינמיים.

כך למשל עבור דף הפרופיל של הילד

\templates\views\kid\profile.hbs

כל הסקריפטים נמצאים ב

\public\js\kid\kid-profile-scripts.js

וכל עיצובי ה css נמצאים ב

\public\css\kid\profile.css

**פירוט חבילות npmתחת dependencies**

* **@google-cloud/storage –** חבילה עזר לשימוש ב Google Cloud Storage.
* **bcrypt–** חבילה להצפנת סיסמאות על מנת לשמור סיסמאות בבסיס המידע בצורה מאובטחת.
* **body-parser –** חבילה שמאפשרת ל express לקבל ולפענח פרמטרים בפורמט JSON שנשלחים ב Ajax מהדפדפן, באמצעות XMLHttpRequest . ניתן לגשת לפרמטרים תחת req.body
* **connect-flash –** מאפשר שליחת הודעות פלאש זמניות למשתמש בתוך Session, אשר נעלמות לאחר הצגתן.
* **connect-pg-simple –** חבילה שנועדה לניהול אחסון .PostgreSQL session
* **cookie-parser –** חבילה לניהול cookies תחת האובייקט req.cookies
* **dotenv -** חבילה שמשמשת לטעינת משתני סביבה מתוך קובץ .env לתוך האובייקט process.env. מאפשר להגן על סיסמאות ופרטים סודיים אחרים וגם מאפשר שימוש במשתנים שייחודיים לכל משתמש.
* **Express –** תשתית המעניקה כלים לפיתוח אפליקציות Web. מאפשרת להגדיר middlewares  עבור בקשות HTTP. מגדירה טבלת ניתוב לטיפול בנתיבים שונים. מאפשרת לרנדר דפי HTML דינמיים על בסיס של העברת ארגומנטים לדפי התבנית.

בעזרת חבילה זו ניתן להגדיר שרת Web שמאזין לבקשות HTTP.

* **express-promise-router –** חבילה שמאפשרת לארגן נתיבים של Express בצורה נוחה. מאפשרת להגדיר נתיבים בקובץ נפרד.
* **express-session –** חבילה שמאפשרת לטפל בסשניםובעוגיות.
* **fs-extra –** חבילה שמאפשרת טיפול בקבצים מקומיים, כגון העברה ממקום למקום.
* **handlebars-helpers – חבילה שמכילה פונקיות עזר רבות עבור handlebars.**
* **Hbs –** חבילה שמהווה view engine של handlebars עבור Express. מאפשרת לרנדר דפים דינמיים.
* **Multer –** חבילה שמאפשרת העלאת קבצים מתוך טפסי HTML.
* **multi-lang –** חבילה שמאפשרת תרגום לשפות שונות. משמשת תרגום בצד השרת בפרוייקט שלנו. הקובץ שמכיל את התרגומים השונים בצד השרת src\language-server.json
* **node-gyp–** חבילה שמאפשרת ניהול גירסאות של חבילות אחרות. הכרחית עבור חבילה אחרת מהרשימה.
* **node-sass–** חבילה שמאפשרת לקמפל קבצי .scss
* **Nodemailer –** חבילה שמאפשרת שליחת אימיילים.
* **Passport –** חבילה שמשמשת עבור אימות משתמשים שמחוברים לאתר.
* **passport-local –** חבילה שמאפשרת אימות משתמשים באמצעות שם משתמש וסיסמה.
* **Path** – חבילה שמעניקה כלים לעבודה עם קבצים ותיקיות.
* **Pg –** לקוח של PostgreSQL .
* **Validator –** חבילת עזר לטיפול במחרוזות.

**פירוט חבילות npmתחת devDependencies**

* **Nodemon –** חבילה שמאפשרת פיתוח נוח ע"י אתחול מחדש של node בכל פעם שקובץ שנמצא במעקב מתעדכן.
* **npm-check –** מאבחן dependencies של חבילות ישנות או שלא תקינות.

**הפרדה בין production לבין development**

על מנת להפריד בין קוד שרץ במצב פיתוח (development) לקוד שרץ באתר (production)

אנחנו עושים שימוש בקבוע שמוגדר בקובץ .env תחת השם NODE\_ENV.

ערכו של משתנה זה במצב פיתוח הוא NODE\_ENV=DEVELOPMENT , לעומת בענן מוגדר אוטומטית המשתנה להיות NODE\_ENV=PRODUCTION.

באופן זה אנו מפרידים בין שני המצבים.

**פיתוח מקומי יחד עם גישה לבסיס נתונים שבענן**

על מנת להקל על הפיתוח והתחזוקה ניתן לגשת ישירות למידע שנמצא בבסיס הנתונים שבענן באופן הבא :

* משנים את פרטי הבסיס נתונים בקובץ .env כך שיתאימו לפרטי בסיס הנתונים שבענן ולא בסיס נתונים מקומי:

יש לשנות את המשתנים: SQL\_USER, SQL\_PASSWORD, SQL\_DATABASE

לערכים זהים לערכים שבקובץ app.standard.yaml

את המשתנה SQL\_PORT יש לשנות ל 5433 (בשונה מהפורט הרגיל 5432)

* פותחים Windows Powershell
* מנווטים אל תיקיית docs שבה נמצא הקובץ cloud\_sql\_proxy.exe
* מריצים את הפקודה:

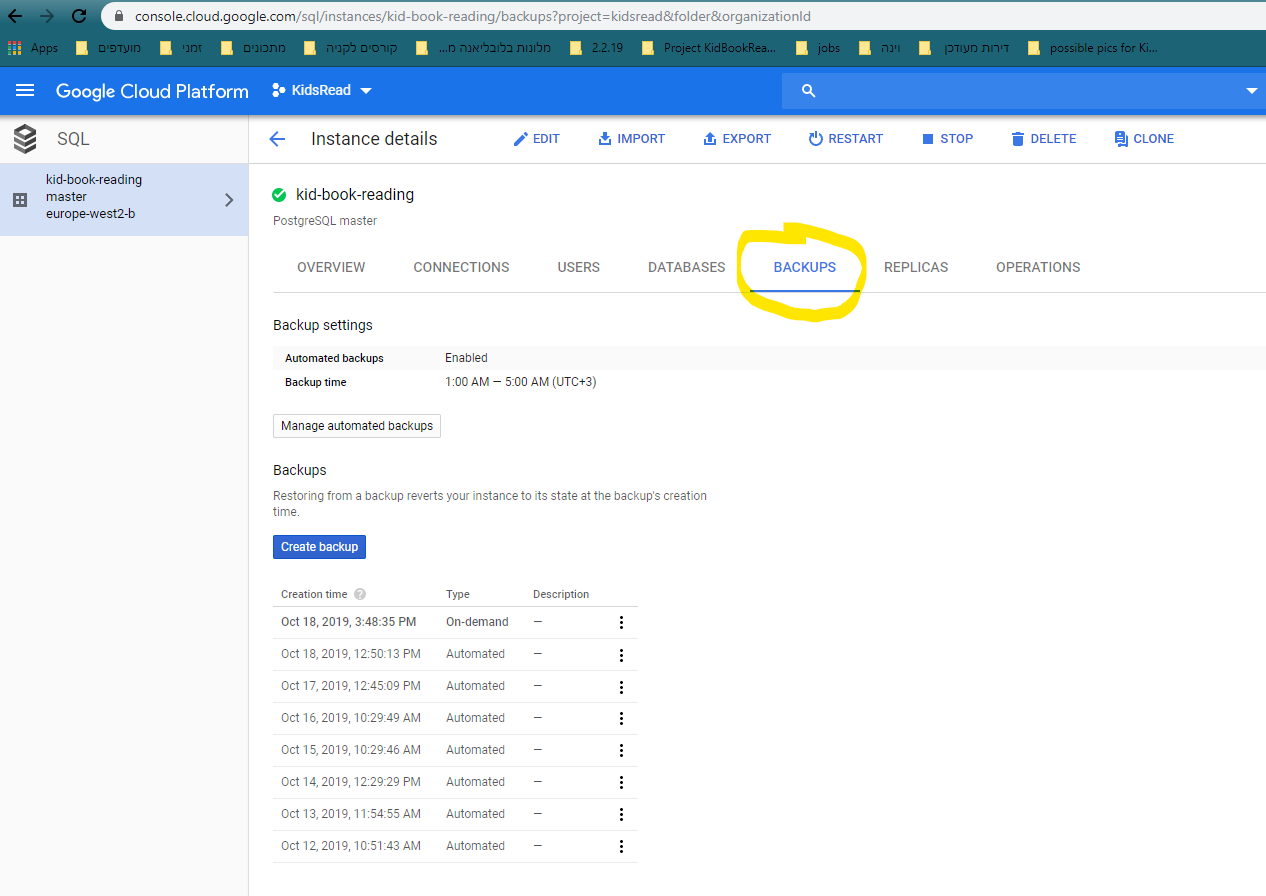
./cloud\_sql\_proxy.exe -instances=kidsread:europe-west2:kid-book-

reading=tcp:5433

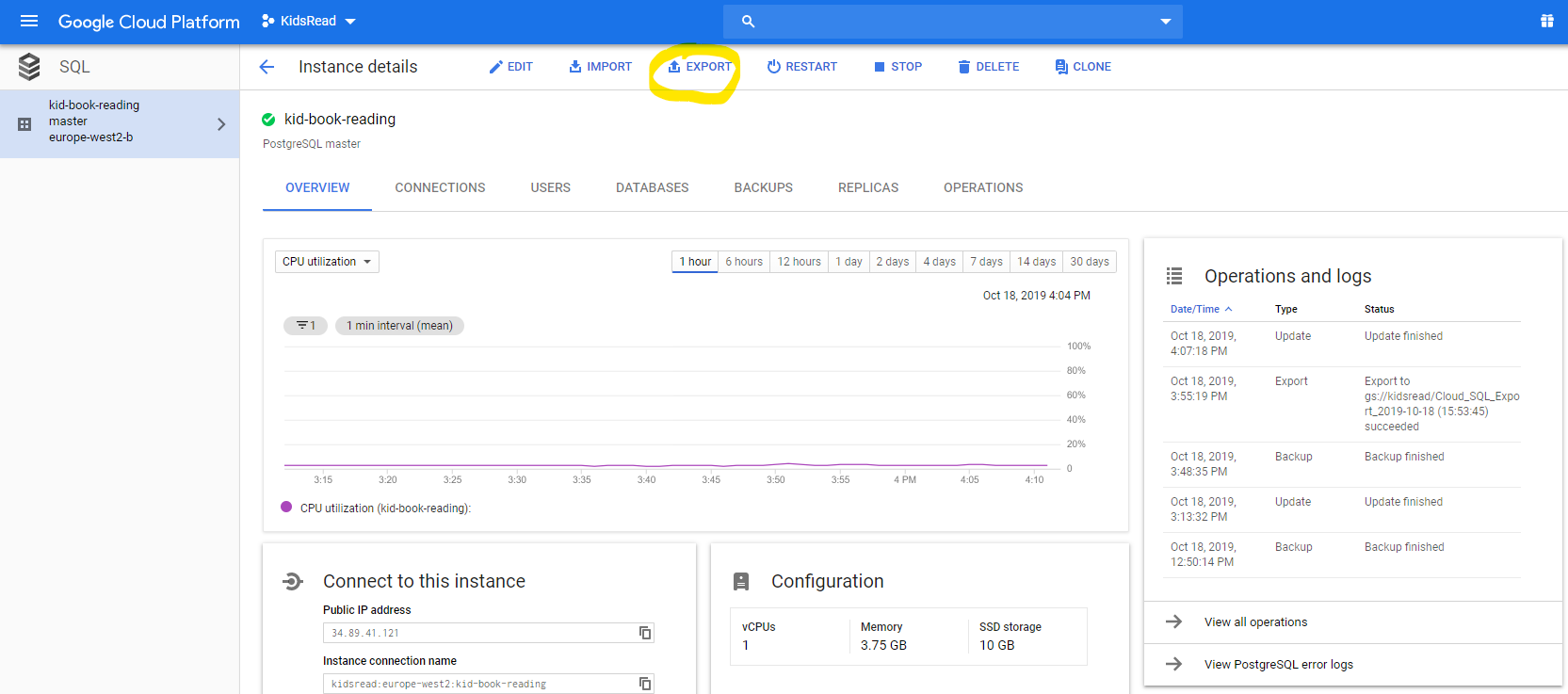
* כעת ניתן להריץ את הפיתוח המקומי בעזרת הפקודה npm run dev, כך מתקבלת גישה ישירה לבסיס הנתונים שבענן בשילוב עם פיתוח מקומי.

**גיבוי בסיס הנתונים ב- Google Cloud SQL**

לבסיס המידע מתבצע גיבוי אוטומטי באופן יומי בין השעות 1:00 – 5:00.

****

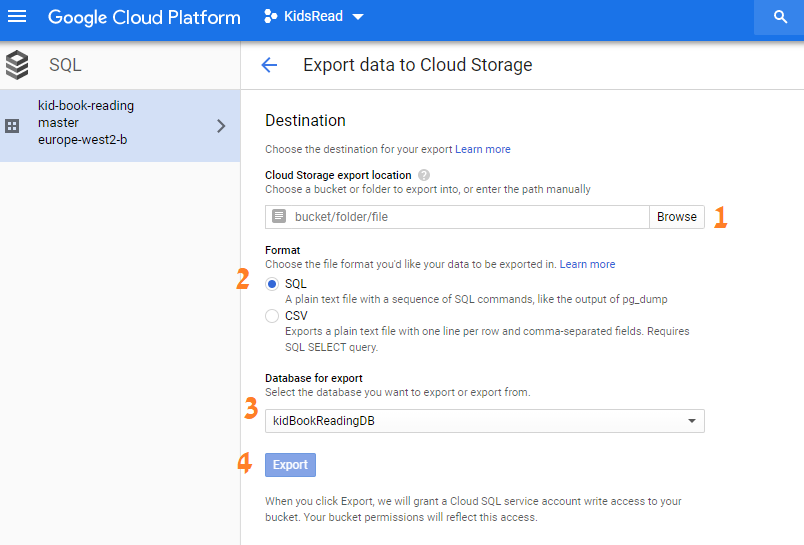
ניתן לייצא קובץ גיבוי של בסיס הנתונים ע"י Export באופן הבא :

* לוחצים על Export :
* 1. בוחרים לאיזה bucket הקובץ יעלה.

2. בוחרים פורמט. SQL מייצא רצף פקודות מלא עבור הבסיס נתונים.

3. בוחרים את בסיס הנתונים.

4. לוחצים Export.



* לאחר מכן ניתן להוריד את הקובץ שנוצר מהמיקום שנבחר.
* למידע נוסף על גיבוי:

<https://cloud.google.com/sql/docs/postgres/backup-recovery/backups>