חלק שני – דרישות ענן

הענן שנבחר הוא GCP

אלו הרכיבים של המערכת בענן:

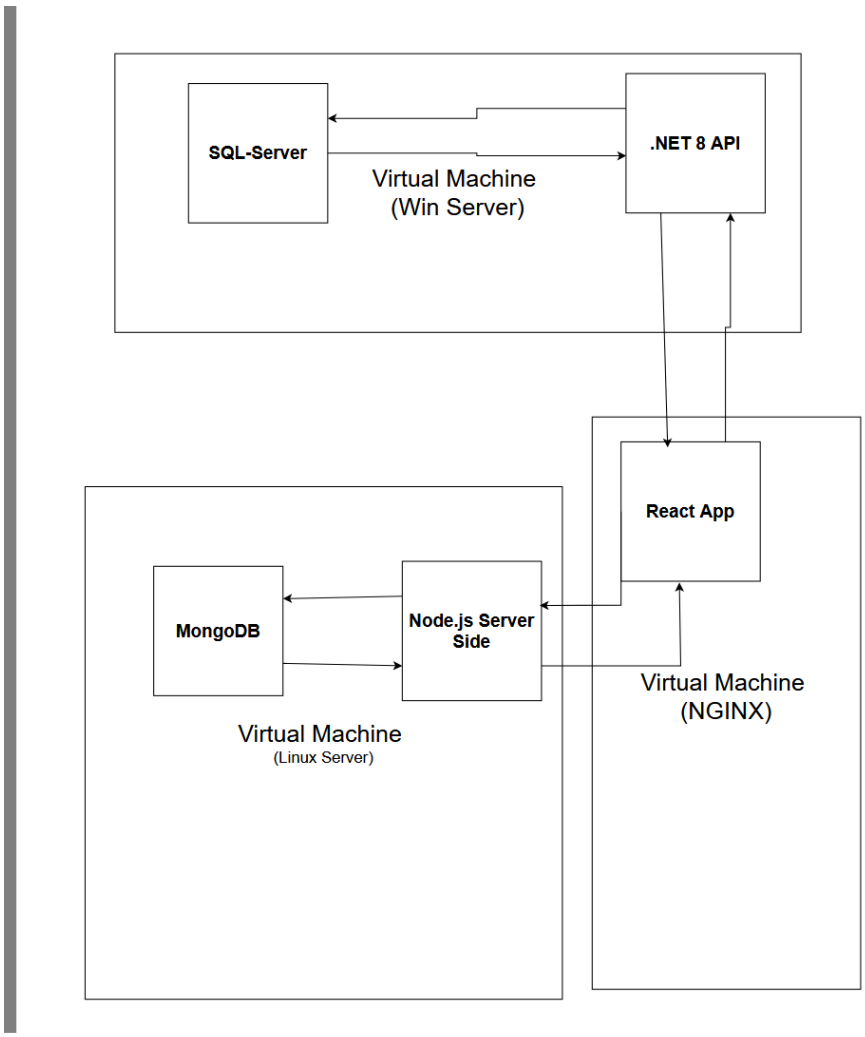
MS SQL Server: שירוץ על מכונה וירטואלית עם מבוססת Windows Server.

אפליקציית ריאקט (React): נריץ למשל על שרת Nginx

צד שרת ב-DOT NET 8: יכול לרוץ על Windows או Linux כאפליקציה.

צד שרת ב-Node.js יחד עם MongoDB – יכולים לרוץ על שרת לינוקס (למשל).

הערה- אפשר היה גם בגישה חלופית במקום מכונות וירטואליות עם צורך יותר גדול בניהול, לעבוד בגישה של שירותים מנוהלים אבל עם פחות שליטה בתשתית.



הסבר התרשים:

את צד השרת ב-.NET יחד עם ה- MS-SQL נריץ יחד על מכונה וירטואלית מבוססת חלונות

נתקין ידנית את SQL Server ונגדיר וננהל את הבסיס נתונים, משתמשים והרשאות.
כמו כן נהיה אחראים על גיבויים, עדכוני אבטחה וכו'.

בנוסף עבור החלק של .NET
נתקין .NET 8. ונפרוס את ה-DLLs ונגדיר את ה-IIS
פריסת קבצי האפליקציה (DLLs, וכו').

את צד השרת ב-.Node.JS יחד עם ה- MongoDB נריץ יחד על מכונה וירטואלית מבוססת לינוקס
נתקין את ה-Node.js כולל ביצוע פקודות npm
נתקין ונגדיר (כולל גיבויים לפי צורך)את השרת MongoDB
נבצע פריסה קבצי האפליקציה של Node.js ונגדיר שרת אינטרנט כמו Nginx עבורו

את האפליקצית ריאקט נריץ עם מכונה וירטואלית נפרדת בשרת מבוסס NGINX (אפשר היה גם להריץ על אחד מהמכונות הקיימות האחרות, רק שהיתה תלות בינהן במקרה של תקלה – אז נבחרה האופציה של מכונה וירטואלית שלישית במספר).

נגדיר שרת **Nginx על המכונה וירטואלית הנ"ל** ונפרוס את הקבצים ע"י npm build ובקשות לצד שרת ינותבו לפי צורך (לכל אחת מהמכונות האחרות – שבכל אחת מהן יש צד שהאפליקציה עושה בה שימוש).

לפי צורך אפשר להוסיף למערכת Load Balancing (תלוי בגודל המערכת) מה שיצריך מכונות וירטואליות נוספות במקרה כזה.

אבטחה
בכל מה שקשור באבטחה נעשה שימוש לשם כך ב-VPC ו-נגדיר כללים (פתיחת פורטים וניתוב התעבורה וכו') ב-Firewall ככה המכונות יתקשרו באופן מאובטח
למשל להגדיר כללים כדי שהרכיבים בכל מכונה יוכלו לתקשר בינהם וכמו כן גם לפתוח פורטים כמו 443 ו-80 לתעבורה חיצונית.

נגדיר הראשות מינימליות לכל משתמש (הכוונה בצורך הכי בסיסי שצריך)
אם נרצה לעשות שימוש ב-SSH אז נגביל את הכתובות המורשות להתחבר רק על מי שמורשים לכך מראש.

אוטומציות ותהליכי DevOps:
נעבוד (למשל) עם GitHub וניצור repo נפרד לכל חלק (.NET הריאקט וה- Node.js) עבור האפל' ריאקט ניצור קבצים סטטיים ע"י npm run build וכמובן התקנת ספריות נחוצות ע"י npm i (גם עבור ה-node.js) עבור ה-.NET ע"י dotnet publish ניצור קבצי הרצה ונגדיר את שרת ה-IIS עבור הקבצים האלו.

נוכל להשתמש בקונטיינרים (דוקר) עבור ה-NET ו-Node.js. עבור כל פעולת Push בריפוזוטרים, נוכל להפעיל אוטומציה שבה נוריד את הקוד ונתקין ספריות נחוצות ונוכל להריץ בדיקות וביצוע בניית הקוד ויצירת תמונה עבור דוקר נבצע חזרה push של התמונות וקבצי ה-build של האפל' ריאקט.

גיבויים:

גיבויים של המערכת – יש לנו קוד מגובה ב-github, לגבי ה-DB יש לנו גיבוי אם נשתמש במונגו Atlas בענן ולגבי ה-SQL נוכל להשתמש ב- Cloud SQL כדי לנהל גיבויים בזמנים מסוימים וכו'.