מסדי ענק חישוב בענן

סמסטר חורף 2021/22

יואל שמלה

רוברט מרטין*

הרצאה 1: 12.10.2021

node js- פלטפורמה ליצירת קוד.
spark- מנוע עיבודים, יודע לתמרן עבודה בין המחשבים ולייעל זמן.
מאגרי נתונים- mongo db, redis

מחשוב ענן- backend
iot- בעיקר ב- frontend
kdd- knowledge discovery database

פיתוח תוכנה מתחלק לשתי קטגוריות:
1. מבנה\ עיצוב קוד
2. חישוביות (אלגוריתמיקה).

עיצוב: המהות של העיצוב זה תכנון ופיתוח של מוצרים תחת אילוצים ודרישות.
האילוצים דורשים מאיתנו להיות גמישים ודינמיים, להיות פתוחים להרחבות וסגורים לשינויים (ocp אבסטרקטי).

ארכיטקטורה- סוג של עיצוב, מושג שיורש מעיצוב.

(שאלת מבחן) ההבדל הוא שבארכיטקטורה השינויים דורשים זמן רב ולפעמים בלתי אפשריים, ההחלטה בארכיטקטורה נוטה להיות כבדה בשינויים שלה.

עיצוב מודולרי- כשרוצים לבנות מערכות נפשוט אותן לחלקים, לחלק את השלם לחלקיו. (עיקרון soc)
מודולו- חלק משלם שהופרד ממנו בצורה נכונה, כלומר חלוקה נכונה והוגנת.

מודולו שחולק בצורה נכונה מורכב משתי תכונות:
1. לכידות- תכונה שיש למודול ביחס לעצמו, עיקרון ההפרדה, כלומר אחריות בלעדית על דבר אחד ככה שאם אותו דבר נהרס רק גורם אחד אחראי לזה.
2. צימוד- תכונה שיש למודול ביחס למודול אחר, מתייחס לסביבה שלו.

לפי רוברט מרטין- עיצוב, זה לומר מי תלוי במי ואיך.

תוכנה- אוסף של שירותים דיגיטליים.
מערכת הפעלה- אוסף של שירותים גנריים כגון: זיכרון, cpu, io
אפליקציות- יישום, אוסף של שירותים ייעודים למטרה מסוימת.
ספריות- אוסף של שירותים, מעין אבני בניין לפיתוח מערכות.
framework- אוסף של יחידות בניין לפיתוח תוכנה.

ההבדל בין ספריות ל framework הוא ש- framework מציע עיצוב מומלץ עם התשתיות

מערכות מבוזרות- מערכות שמרכיביה נמצאות ביותר ממקום אחד פיזית.

לביזור יש רמה זה נקרא tier
במערכות tier 2 החלוקה היא ל client ו server, בשנות ה- 90 החלוקה הייתה ל- tier 3,
כיום החלוקה היא ל -n tier.

החלוקה היא ל- *froentend* ו *backend*:

froentend: client side

backend: שרת, מחשב שנותן שירותים לאחרים. נתעסק בעיקר ב**שרת אפליקציה** (שרתים שעושים עיבוד) ו**שרת נתונים** (דואג לטפל בנתונים בצורה יעילה data base).

מחשוב ענן- צורת מחשוב שמתבססת שמישהו נותן לי שירות מרוחק, לעבוד עם ספקי שירותי ענן.
רשת- שיטה לשיתוף משאבים. (דוגמא למשאב יכולת איחסון).
service- אבולוציה של אובייקט.
מה שרלוונטי לי בשרת זה שאפשר לשוחח איתו, לא רלוונטי השפה והצורה וכו׳.

scale: תכונה של מערכת, יכולת של מערכת תוכנה להתאים את עצמה לשינויים בנפח עבודה נדרש בשינויי חומרה בלבד וללא שינויי תוכנה.
scale up: הרחבת גודל הפס
scale over: לפזר את התוכנה בכמה מחשבים ופיצול העומס, כלומר ניתוב
-כשמרבים ב scale up מגיעים לצוואר בקבוק, העדיפות היא להרבות ב scale over.

dev ops- שילוב של עולמות פיתוח תוכנה עם תשתיות.
וירטואליזציה- תופעה במציאות שהוצאנו לו אילוץ פיזי.

מטרת הקורס- טיפול בנתונים בסיטואציות מיוחדות

מודל dikw: מתייחס לסוג של מודלים לייצוג מבנים או פונקציונליים.
Data **I**nformation **K**nowledge **w**isdom

מודל:DIKW
זוהי פירמידה שמורכבת מ-4 רמות, המייצגות כל אחת מהן רמת מורכבות גדולה יותר של ידע והבנה :
Data, Information, Knowledge, Wisdom
המודל הזה משמש גם לצורך ניתוח וחקר של תהליכי למידה וידע אנושיים וגם כדי לתת מענה לצרכים של ניהול ידע ארגוני.

Data (נתונים)- הם המידע בצורה הכי מולקולרית שלו. סמלים, מילים וערכים מספריים או מדידים שונים . (אוסף של נתונים שאינם מאורגנים או מסודרים).

Information (מידע)- הוא אוסף של נתונים המאורגן בצורה שנותנת לו משמעות.
מידע יכול לענות על שאלות של "מי", "מה", "איפה", "כמה" ו"מתי״.

Knowledge (ידע)- הוא מושג מעורפל שקשה יותר להגדיר. הוא כולל שילוב של פריטי מידע.
אולם זה מתקיים בתוך מסגרת של ניסיון, עולם ערכים, הקשר, תובנות ואינטואיציה, אשר בתוכה מתקיימים ההערכה והשימוש במידע. (אנליזת מידע).

Wisdom (חוכמה)- הוא המושג המעורפל ביותר. חלק מהעוסקים בתחום כוללים בתוכה רמה נוספת שהיא הבנה (understanding), המוגדרת בתור היכולת להעריך מתוך הידע גם את ה"למה" וכן את היכולת לייצר על סמך הידע, פריטים חדשים של מידע. החוכמה היא מה שנבנה על בסיס ההבנה. זהו המקום שבו אנחנו מעריכים, שופטים ודנים בהיבטים שונים של הידע שלנו. זו היכולת שלנו להבחין בין טוב לרע, לשאול שאלות חדשות ולייצר הבנה וידע גם על בסיס מידע חלקי.

(במבחן) אונתולוגיה- ייצוג פורמלי של מושגים מאיזשהו דומיין והיחסים ביניהם.

הרצאה 2: 19.10.2021

big data מתחלק לשני צירים:
ציר אחד: data base, כלים ושיטות לטיפול בנתונים, machine learning
ציר שני: מחשוב ענן- קונספטים, עקרונות, כלים, טכנולוגיות וכו׳

-כל רכיב צריך קישור טוב לאינטרנט על מנת לקבל את השירותים. כמו שצריך.

software services- יחידות קצה שמהן נבנה אפליקציות, קומפוננטה חישובית שנגישה מכל מקום לצרכן, מודל חישוב שיעיל מבחינה פיננסית.
ipc: inter process communication
לדוגמא ctrl c ו ctrl v

מחשוב מבוסס דוקומונטיים- מבוסס על העברת html,
json- מסמך טקסט שעובר בין מחשבים.

(שאלת מבחן)מה ההבדל העקרוני בין xml, json, csv?
כולם מעבירים נתונים, רמת הדחיסות ורמת הביטחון הן שונות:
xml מקסימלי
csv מינימלי
json באמצע.

-בעבר, הפונקציות היו הבסיס לפיתוח, לאחר מכן המחלקות, לאחר מכן קומפוננטות ולבסוף software service

שירות תוכנה- נותן או איחסון או עיבוד, יכול להיות במסגרת ענן או לא.
ענן- אוסף של שירותים ורמת גרנולציה(התבוננות במאקרו או מיקרו).

מאקרו- paas, saas -ספריה של הרבה שרתים שבודקת מה רצון הלקוח:

iaas- מכונות וירטואליות, אחסון, נתבים וירטואליים
paas- תשתיות לפיתוח אתרים, אירוח שירותי ענן, אחסון וניהול נתונים (טבלאות נתונים DB , Blobs , רלציוני), שירותי מסרים.
saas- Goolge Docs, Office 365.

מיקרו- web service, אם שמים אותו תחת שרת שבאחריותי, אני צריך לדאוג לאבטחה שלו במידה ושמתי אותו בספק של שירותי ענן אני מסיר ממני את אחריות הטיפול, שזה עצם ההבדל בין *web service* לבין *cloud service*. כלומר תחומי אחריות(שאלת מבחן).

מונולינט- בעל צימוד גבוה, לא יעיל משום קישוריות רבה שפוגעת בתחזוקה(הרחבה וכו׳)
domain logic- מחייב אותנו להפריד בין האלמנטים כלומר אי תלות, הפרדה.
מודול גישה לנתונים- השכבה שאחראית להביא לנו נתונים מהקוד.

תבנית השכבות:
פרזנטציה- (*השכבה עליונה*) תצוגה, תלויה בשכבה שתחתיה
לוגיקה מרכזית- (*השכבה האמצעית*) domain logic, תלויה בשכבה שתחתיה ולא מעליה
גישה לנתונים- (*השכבה התחתונה*) מודול גישה לנתונים, כלומר לשאוב נתונים כגון data base וכו׳
כל אחד הוא מחלקה כי הן קומפוננטות.

S.O.A -service oriented architecture עיצוב של מערכות מידע בתוך ארגון, כלומר ביסוס מחשוב אירגוני.
esp- מנגנון לחיבור השרתים.
http- פרוטוקול תקשורת.
מחלקה- היא תבנית לאובייקט
service- הוא סוג של אובייקט.

microservices- ארכיטקטורה אבן בניין בפיתוח אפליקציות בסביבת ענן שמאופיין במיקוד פונקציונלי, מבצע אך ורק פעולה אחת בצורה מאוד טובה.

information barriers- כל אחד אחראי לתחום אחריות שלו בלבד.
טרנזקציה- יחידה אטומית לביצוע משימה.

עיקרון עיצובי לטובת פיתוח תוכנה – IOC (inversion of control).

הרצאה 3: 26.10.2021

rest- סוג של ארכיטקטורה, כאשר מחברים דברים ביחד צריך לתת לכ״א id.
http הוא מימוש פרטי של rest.

סגנונות של עבודה:
statelessly – *פונקציה* שמבצעת פקודות **ונמחקת**. (http)
statefull- *אובייקטים* שמתקשרים אחד עם השני ע״ העברת סיגנלים.

cookies- מנגנון של state managment. (נמצא אצל ה client)

(למבחן)
chat- תקשורת רבה אבל עם קצת נתונים, מתאים ל- statelessly
chunky- העברת **כל** חבילת הנתונים(פיסה, העברה של הכל), מתאים לstatefull
(מעבירים את כל הנתונים), כלומר בכל ממסר שיעבור, כל הנתונים יחזרו שנית.

api- סוג של ממשק, interface.
technology stack- אוסף של טכנולוגיות שמביאות פתרון שלם.
אובייקט- אוסף של דברים.

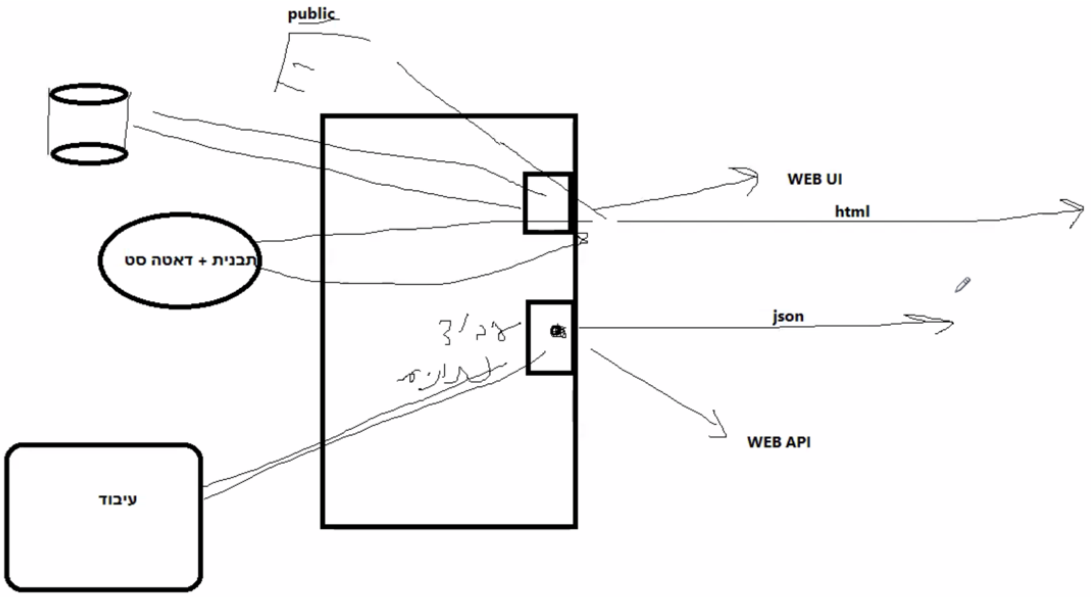
intent- כוונה

הרצאה 4: 2.11.2021

web service אני אחראי על הטיפול במעטפת
cloud service אני מטפל בלוגיקה בלבד וכל המעטפת אני לא אחראי לזה

השרת מספק לנו web ui ו- web api:
api- עיבוד נתונים, יודע לעשות חישובים שאני לא יודע, מבצע עיבוד ומחזיר לנו בד"כ טקסט (json, xml)

ui- בד"כ מחזיר html, יש לו שני דרכים:
1- לוקח מספריה (כמו public)
2- עושה שימוש בתבנית + שימוש ב-data set שמייצרת שימוש ומוציאה דף html.



routing- שיטות לאיך קוראים לפונקציה ואיך מעבירים נתונים

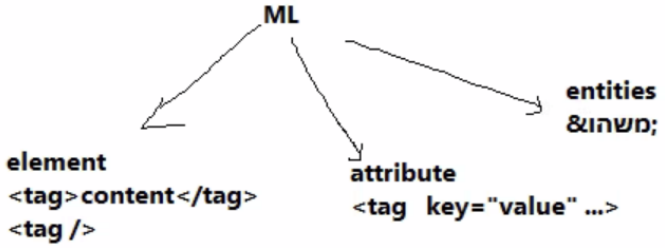
url Uniform Resource Locator – כתובת בזיכרון שאני פונה אליה ומקבל בצורה יחידנית משאב.

query string- סטרינג שמוסיפים ב url אחרי הסימן שאלה והוא מכיל key ו- value

parser- ניתוח נתונים, שיטה שבה מחרוזת נתונים אחת מומרת לסוג אחר של נתונים. נניח שמקבלים את הנתונים ב-HTML גולמי, ה- parser ייקח את ה-HTML הנ"ל ויהפוך אותו לפורמט נתונים קריא יותר שניתן לקרוא ולהבין בקלות.

url encoding- ממיר תווים לפורמט שקל לעבוד איתם.
url decoding- המרת תווים, פענוח כתובת האתר היא הופכת את הקידוד.

- מסמך טקסטואלי בנוי משלושה מרכיבים:
1. אלמנט
2. attribute
3. entities, פותר בעיות של כתיבה



sgml- הוא ה- "אבא" של xml ו html.
xml הוא תקן כבד, לכן כיום לרוב משתמשים ב json.
html תפקידו לארגן תוכן ולהעביר נתונים לשרת, הוא לא אחראי על התצוגה(!).

הרצאה 5: 9.11.2021

<u>איך מארגנים את הקוד באמצעות mvc ל- server</u>

משימה בנויה משלושה חלקים:
-לוגיקה
-תצוגה
-נתונים

נארגן את הקוד למשימות כך שלכל משימה יהיה mvc.

Mvc (model, view, controller) - תבנית עיצוב ארכיטקטונית שמשמשת לארגון קוד המשמש תכונות רצויות, כגון: שימוש חוזר, תחזוקתיות, הרחבה וכו׳.
-כל אחד מהחלקים ב mvc הוא מודול (חלק משלם).

model- קוד שזוכר נתונים משפץ אותם, מתחזק אותם וכו׳
טכנית, קובץ שיש לו קוד ובסוף export, לבסוף מייצאים איזשהו אובייקט.

view- תצוגה, מנוע מחולל תצוגה, פועל בדפים דינאמיים.

Controller- בקר, מנהל הממשק, תפקידו הוא לקשר בין ה model ל view,
איך זה נעשה בפועל? ה- view מקבל קלט מהמשתמש (לדוג׳ לחיצה על כפתור באפליקציה) ה- view מיידע את Controller על מה שאירע, ה- Controller מעביר את הנתונים ל-model והוא מעבד אותם (לדוג׳ מוסיף סטודנט לרשימת התלמידים של מורה מסוים).

front controller - נתב, מקבל את כל הבקשות ומנתב למשימה ספציפית.

tangel -כאוס, מעין חוטים שמסתבכים, עקב תלות בגורם יחידני.

login- תיעוד פעולה כלשהי ביומן, על מנת לבצע מעקב.

pipes and filters – תבנית עיצוב, יש לה יכולת לחבר בין אובייקטים, היא מאפשרת זרם או עיבוד א- סינכרוני.

middleware- מתווך, פיסת חומרה או תוכנה אשר נמצאת בין ה- server ל- client ולא בתוכם(!), עומד בין הבקשות לשירות (proxy הוא דוגמא ל- middleware).

kafka- משמש כ- middleware, תשתית שמשמשת לעיבודי ענן, נלקחת מהספקים.

הרצאה 6: 16.11.2021

cross cutting concern – עיצוב קוד נכון, aspect מסוים בתוך הקוד שאי אפשר להפרידו בצורה טובה משאר ה- concern(נושא, אוסף נתונים והטיפול בהם).

scattering- פיזור, קוד מפוזר ולא מסודר.

-התשתית של express עוברת דרך ה- midleware על מנת לוודא הרשאות לשרת, תהליך זה יקרה לפני המעבר לפונקציות.

-לכל controller אפשר לשים לפניו פונקציות של midleware.

-האובייקטים router ו controller זהים, router יכול להכיל midleware.

-ה pipe מה- front control (או אפילו מתת control) ל –mvc, מכיל בתוכו midleware.

הקינפוג של midleware- ע״י app, use מעבירים בין המצביע לפונקציה.
התבנית של midleware היא respone, request, next
יוצרים פונקציה שמחזירה פונקציה, היא מקבלת את הפרמטרים ומריצה אותם בפונקציה אחת.

קלוזר׳- רעיון תכנותי ב- java script של פונקציה שזוכרת נתונים(לא טריוויאלי שהפונק׳ הפנימית תזכור את options).

פרוטוקול WEB socket- פרוטוקול full duplex, בו לא רק הקליינט יכול ליזום תקשורת עם השרת, אלא גם השרת יכול ליזום עדכונים ותקשורת לקליינט.

view engines - פקדי שרת אשר פועלים בתוך האפליקציה לעיבוד תצוגה לדפדפן או למשתמש.

תשתית טובה עובדת עם מוסכמויות, ישנה עדיפות על פני קונפיגורציות.

ejs מקבל סט נתונים ותבנית, מסדר בצורה דינמית ושולח ל client או ל server כרצונו.

simpso'n paradox: מראה איך ניתן לקחת את אותם הנתונים בדיוק ולקבל מסקנות או החלטות שונות לגמרי, 2 ההחלטות הינן רציונליות ושונות משום שבצועה חלוקה שונה בהסתברות מותנית.
נובע מכך שבהרבה מקרים חסר פרמטר שיגרום להחלטה להיות נכונה.

הרצאה 7: 23.11.2021

data base- מסד נתונים, סוג של תוכנה שהייעוד שלה זה לאחסן ולשתף נתונים עם אופטימיזציות שונות. בד"כ מותקנת על מחשב המספק שירותים (שרת נתונים).
– rdbms
-sql

NoSQL:
1. **key value DB** - מסד נתונים שמייצג את השיטה נקרא redis, ניתן לעבוד איתו בענן, מיכל חכם הנקרא docker, או ישירות במחשב.
שיטת עבודה "רגילה" נקראת **OLTP** שייכת למערכות תפעוליות

-כשעובדים עם *רלצ'וני* - נחבר כמה מחשבים יחד על מנת להגביר תפוקה אך יש לנו חסם כלשהו ולכן התועלת לא תהיה מירבית עקב הנירמול.
לעומת זאת העבודה עם *redis* יעילה כי היא יודעת לקרוא למחשב הנכון בזמן הנכון ולייעל את התפוקה.

2. **document** – מסדי נתונים ללא טבלאות ורשומות, מכיל את כל הנתונים של ישות מסוימת.
mongo db – מקרה קלאסי של document.
לכל document יש ip

3. **graph**- מסדי נתונים שמייצאים נתונים כגרפים. (לדוגמא waze),
neo4j ייצוג של מסד נתונים זה.

cap theorem - כשעובדים עם מערכות מבוזרות, צריך להתפשר על מה מספקים יותר ואי אפשר להבטיח יותר משתיים מבין שלוש התכונות הבאות:

1. *consistency* **–** עקביות, , בכל קריאה מבסיס נתונים מתקבל המידע העדכני ביותר

2. *availbaility* – זמינות, בכל קריאה מבסיס נתונים מתקבלת תשובה

3. *partition tolerance* – יכולת חלוקה, סבילות

OLTP: סוג של עיבוד שעוסק בתפעולים, מיוצגות כטבלה, לוכד מאחסן ומעבד נתונים בזמן אמת
OLAP: סוג של עיבוד שעוסק בניתוחים, מיוצגות כקובייה, משתמש בשאילתות מורכבות כדי לנתח נתונים ישנים.

טרנזקציה- פעולה לוגית אחת, תנועה.
לדוגמא העברת כסף בין חשבון בנק לחשבון אחר פעולה זו, למרות שהיא כוללת מספר פעולות בדידות (חיוב החשבון המשלם וזיכוי חשבון הנמען), יש להתייחס אליה כפעולה לוגית אחת.
תכונות ה- acid דואגות לכך שהפעולה תעבורנה באופן אמין.

כדי שטרנזקציה תהיה עקבית מפעלים ארבעה דברים:

- אטומית A
- עקבית C
- מבודדת I
- עמידה לפגיעות D

index - מנגנון שיודע לגשת לנתון ספציפי ולשלוף אותו.

mongo db – ישנם שני סוגי עבודות:

- עבודה עם סכמה
- עבודה בלי סכמה, בעולם של ביג דאטה, לא תמיד יש זמן להתעסק בתחזוקה אלא קודם כל נשמור אותן.

ל mongo db יש דרייבר שנקרא mongos

הרצאה 8: 30.11.2021

collection - אוסף של document
message- ישות של נתונים.
מודל – ייצוג מופשט של המציאות

restful api – סגנון תוכנה ארכיטקטוני על מנת לממש שירותי רשת, העיקרון הוא להגדיר משאב שהמצב שלו עובר שינוי כתוצאה מהאינטראקציה בין הספק לצרכן.

graph ql- אלטרנטיבה ל- **restful api**, דורש את התבנית של התשובה מראש, צריך לדעת את ה- type שלו, מה שמהווה חיסרון.

מערכת מונוליטית- מערכת שלא מפורטת לחלוקת עבודה, כלומר אין חלוקה לתחומי אחריות.

daemon- **LINUX** תוכנה שרצה ברקע, יש לה בד״כ עדיפויות גדולות יותר (high priority)

BIG DATA- אוסף של בעיות שקשורות לאחסון נתונים בסיטואציות אינטנסיביות.

הרצאה 9: 7.12.2021

edge compute- חלק מהשירותים שעוברים דרך מחשב בדרך לפני ענן.

ספקי שירותי ענן- נותני שירות.
לדוגמא, aws נותנים שירותי חישוב, שירותי database וכו׳

ההבדל בין virtual machine לשירותי חישוב:

הוא ברמת השירותים שמספקים, כלומר
שירותי ענן מספקים, חישוב או איחסון (ברמת המתכנת),
סוגי שירותים שניתן לקבל ב- **vm** (בגדול zoom-out), לדוגמא: remote desktop

mediator- הוא סוג של observer, עושה כימוס בתקשורת בין אובייקטים.

Big Data:

- קיימים המון נתונים זמינים
- ישנן יכולות חישוב
- מטרות: לדעת, להסביר ואף לחזות
- האתגר: לקחת נתונים ולהפיק מהן תובנות, *תובנה אנליטית*.

נושאים הקיימים בתחום: עיבוד מקבילי, AI, סטטיסטיקה, מערכות מבוזרות וכו׳.

Structure- נתונים מסויימים שיש את הסכימה שלהם.
Data Lake- מאגר, מערכת נתונים ששומרת את **העתק** הנתונים הגולמיים של נתוני המקור.

BI- בינה עסקית, עוסקת בבניית מערכות שעוזרות לארגון להפיק מידע משמעותי מבחינה עסקית מתוך נתונים הנאספים על ידיו.

הרצאה 10: 14.12.2021

שאלות מבחן

שיטת עבודה עם מערכת תפעולית- בעבר שמרו נתונים כחלק מתפעול, כלומר מישהו ביצע עבודה נוצרו הנתונים ואז הם נשמרו לשימוש עתידי.
מדד התפעול נקרא OLTP כך שטרנזקציה הינה פעולה שלמה שמכילה תתי פעולות כך שכולן מתבצעות. אותם ביצועים נמדדים ב tps-ים.

TPS- (**transactions per second**) מספר ייצורים שהמערכת עושה בשנייה,
בהמשך חשבו שניתן להסיק מסקנות מנתונים אלו ולראות מגמות.

הבדלים בין מערכת תפעולית למערכת אנליטית:

מערכת תפעולית -מערכות מידע שעוזרות לאנשים לסייע לעשות את עבודתם.
מערכת אנליטית- להפיק מהם תובנות.

הבדלים בין big data לבין small data:

big data	small data
לקיחת **כל** המידע	שאיבת נתונים שהארגון מפיק
מצטבר לאורך זמן לא ידוע	נאגר ורלוונטי למספר שנים
מגיע מכל סוגי רכיבים ופורמטים שונים, ביזור	מגיע מפורמט אחד
אין מחיקת נתונים!	
לא מבינים תמיד מה כל מידע אומר	יודע בכל פיסת מידע מה יש ספציפית
צורך בהכנה מעמיקה מראש	ניתוח בבת אחת
מבניות מעורבת	מובנה
אין אפשרות לשחזר, אם אין תיעוד באותו הרגע	שחזור נגיש

מודל ה vvv:

מודל שמצביע באילו מקרים נתמודד עם big data

volume- מקרים של נפח גדול

velocity- מהירויות וקצבים גבוהים

variety- קבלת נתונים מכל מיני תצורות

veracity- סיווג הנתונים לרמות אמון, מודלים (עקב קבלת מידע עצום).

hardoop- (דיסק קשיח) hard disk וירטואלי שהוא הבסיס של data lake.

הרצאה 11: 21.12.2021

browser - מכונה שיעילה לאינטראקציה עם המשתמש.

שלושת המנועים שבעזרתם מייצרים תצוגה ואינטראקציה עם המשתמש:

- css
- html
- java script

www כתובת הראש של אתר האינטרנט (דפי טקסט) - שירות שמבוסס על hyper media.

ajax-שיטת עבודה, לקריאה ישירה להבאת דפים וכל סוג של קובץ כולל json וכו׳.

cdn- שרתים שפתוחים לציבור וניתן להוריד מהם דברים.

הרצאה 12: 28.12.2021

ישנן 2 פונקציות נפוצות ב json:

- הופך אותו לאובייקט ()**json.parse** דיסארליזיציה הפוך מסריאליזציה (שבו הופכים אובייקט ל-json)
-

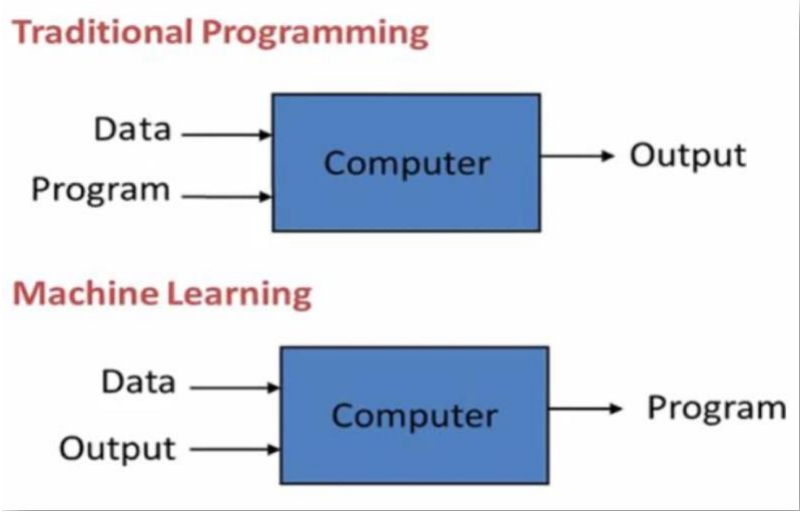
נתוני עתק: nosql, data lake

Data Pipeline- נתונים שנכנסים ממקום מסוים ויוצאים ממקום אחר ובין לבין נמצאים ב "תחנות".
ישנם הרבה מקורות של data שמגיעים ממגוון של מקומות.

HDFS- מנגנון שמייצר hard disk וירטואלי, על מנת לשמר את הנתונים במאגר.

spark הוא מנוע עיבודי, kafka הוא message broker.

ההבדל בין macine learning ל- תוכנית קלאסית:



הרצאה 13: 04.01.2022

רגרסיה ליניארית- למידה מפוקחת. הוא מודל כי מחבר בין ה- packages ולא מתייחס לאף דבר אחר.

עץ החלטה- כל רמה בעץ מורידה את אי הודאות, ככל שנרד יותר בעומק העץ נדע יותר.
אין אפשרות להעלים את האי ודאות ב- 100%

confusion matrix – מתארת את הביצועים של המודל על קבוצה של נתוני בדיקה שערכם האמיתי ידוע.

descriptive model - מודל שעושה חיזוי.

predictive model – מודל שממליץ על כמה אופציות עתידיות.

event - תבנית שמעבדת מעברים(דלתא) ובכך אפשר להגיע למספור הפעולות, כלומר סוכמת איבנטים.

lambda architecture- תבנית ארכיטקטונית לפיתוח של מערכות big data analytic.

stream – זרם של נתונים, מנגנון שמאפשר העברת נתונים בצורה ממושכת.

kappa – בקשת עיבודים בזמן אמת, כלומר רק את ה- hot path.

Serverless- מודל ביצוע מחשוב ענן שהוא הספק מקצה משאבי מכונה ודואג לשרתים בשם הלקוחות, כלומר מסתיר מידע של איך וכמה משתמשים נמצאים במשאבים ומספק רק את המשאבים הנחוצים.