מבוא למחשוב ענן - סמסטר אביב תשפ"ה

תרגיל בית 2

**מגישים: קבוצת "קקדו"**

עומר לב | ת"ז 209403427

מתן ראובן טל | ת"ז 208017772

ספיר גרסטמן | ת"ז 323116186

אריאל לניאדו | ת"ז 318930393

נועה שיטרית | ת"ז 206943219

טל יגודין | ת"ז 324256536

**קישור לעמוד GitHub:**

<https://github.com/Method-for-Software-System-Development/Cloud_Computing>

**מהנדס המערכת:** מתן ראובן טל

**הוראות להפעלת המערכת**

1. **כניסה למחברת הראשית:**

יש לגשת לתיקיית gui ולפתוח את הקובץ dashboard.ipynb.

קישור ישיר: <https://github.com/Method-for-Software-System-Development/Cloud_Computing/blob/main/gui/dashboard.ipynb>

1. **הרצת שלב ההתקנה וההכנה:**

יש להריץ את תא הקוד הראשון תחת הכותרת:SETUP FOR DASHBOARD

שלב זה כולל התקנת ספריות, ייבוא מחברות לוגיות, חיבורים ל- Firebase ועוד...

**התהליך נמשך כ־2 דקות.**

בסיומו, יש לוודא שמתקבל הפלט:

Setup completed successfully.✅

1. **הרצת המערכת:**

יש להריץ את תא הקוד השני. תוך מספר שניות תופיע האפליקציה בתחתית המחברת.

כברירת מחדל, הממשק מוצג בתוך המחברת (בהתאם לדרישה), אך לחוויית שימוש טובה יותר במסך מלא, ניתן לפתוח את המערכת בקישור חיצוני בלבד (ללא הצגה בתוך הקולאב) באמצעות הקוד הבא:

במקום demo.launch() שנמצא בשורה האחרונה בקוד יש לכתוב/להדביק:

demo.launch(share=True, inbrowser=True, inline=False)

**הערה:** לא מומלץ להפעיל את המערכת גם בתוך המחברת וגם בקישור החיצוני במקביל- הדבר עלול לגרום לתקלות בהצגת הנתונים או בקבלת קלט.

1. **נתוני התחברות לבדיקה:**

שם משתמש: matan\_tal

סיסמה: 1234

**חלק 1: תהליך העבודה בצוות, חלוקת משימות ובניית אינדקס**

בפתיחת התרגיל קיימנו פגישת צוות בהשתתפות כלל חברי הצוות, במסגרתה בחרנו במתודולוגיית עבודה לפי עקרונות Agile-Scrum, כאשר מהנדס המערכת ישמש גם כ-Scrum Master.

חלוקת התפקידים בוצעה מתוך ניסיון להתאים בין תחומי העניין והחוזקות של כל חבר צוות לבין הדרישה לשנות תפקיד ביחס לתרגיל הקודם. קבענו מראש שתי פגישות עדכון סטטוס שבועיות לצורך מעקב אחר ההתקדמות, פתרון תקלות וביצוע אינטגרציה בין הרכיבים השונים של המערכת.

כחלק מאסטרטגיית העבודה, כל חבר צוות עבד על מחברת קולאב נפרדת, בהתאם לתחום אחריותו. גישה זו אפשרה עבודה מקבילית ומנעה קונפליקטים בקוד. לקראת סיום, ביצענו אינטגרציה של כל המחברות לתוך מחברת ראשית אחת, הכוללת את ממשק המשתמש (GUI) ומנגנון ההרצה המרכזי של המערכת.

בפגישה המסכמת בוצע תהליך תיקון באגים אחרון לפני ההגשה. בטבלת חלוקת המשימות ניתן לראות כי רוב המשימות הושלמו בהצלחה. נותרו מספר תקלות נקודתיות, אשר לדעתנו נובעות ממגבלות ספריית Gradio בה השתמשנו ליצירת הממשק. אנו ממשיכים לבדוק את הסוגיות הללו לעומק.

**להלן חלוקת העבודה בצוות:**

|  |  |  |
| --- | --- | --- |
| שם חבר הצוות ותפקיד בתרגיל זה | משימות שהוקצו | משימות שהושלמו |
| מתן ראובן טל- מהנדס המערכת ו- Scrum Master | * להוביל את תהליך העבודה לפי עקרונות Agile-Scrum. * לתאם בין חברי הצוות ולוודא שכל אחד מבין את משימותיו. * לקבוע ולהנחות פגישות סטטוס קבועות. * לוודא שהתהליך מתנהל בצורה יעילה ושיש עמידה בזמנים. * לשמש כממשק בין הצוות לבין המרצה (במידת הצורך). | כל המשימות הושלמו בהצלחה בהתאם להגדרת התפקיד. לא נדרש מענה מצד המרצה במהלך העבודה. |
| ספיר גרסטמן- Product Manager | * וידוא עמידה בדרישות שהוגדרו בתרגיל הקודם. * יישום פיצ'רים בהתאם לצרכי המשתמש. * התאמת המערכת לעקרונות האב-טיפוס שגובש בשלבי החשיבה העיצובית. | המשימות הושלמו בהצלחה. הביצוע דומה לאב-טיפוס, למעט התאמות שנעשו עקב מגבלות המידע הזמין מהענן. לדוגמה, ויתרנו על הצגת התהליך שרץ במכונה, מאחר והמידע הנדרש לכך לא קיים בפועל. |
| אריאל לניאדו- Backend Developer | * הקמת מסד נתונים בענן (Firebase). * פיתוח שכבות לוגיות הכוללות התחברות משתמשים, הפקת גרפים סטטיסטיים, וסימולטור תקלות מבוסס נתונים. | המשימות הושלמו בהצלחה. |
| נועה שיטרית- QA | * ניהול מעקב אחר תקלות באמצעות מסמך משותף. * ביצוע תיקוני באגים. * למידת סביבת Gradio לצורך הבנת מגבלותיה והשפעתן על תהליך הבדיקה. | מרבית התקלות תוקנו במהלך הפיתוח. נותרו מספר תקלות עיצוביות מינוריות, אשר ככל הנראה נובעות ממגבלות בספריית Gradio עצמה. |
| טל יגודין- UI | * שיפור העיצוב הראשוני בהתאם לאב-הטיפוס. * בחירת פלטת צבעים מתאימה למערכת. * יישום עיצוב באמצעות כתיבת קוד CSS מותאם אישית. | המשימות הושלמו בהצלחה. |
| עומר לב- Frontend Developer | * פיתוח ממשק המשתמש (GUI) באמצעות מחברת ה-Dashboard הראשית, הכוללת את הצגת הנתונים החזותיים והתפעוליים למשתמש הקצה. | המשימות הושלמו בהצלחה. |

בניית אינדקס

להלן רשימת ה- stop words שבחרנו:

1. about

2. above

3. after

4. again

5. against

6. ain

7. all

8. am

9. an

10. and

11. any

12. are

13. aren

14. aren't

15. as

16. at

17. be

18. because

19. been

20. before

21. being

22. below

23. between

24. both

25. but

26. by

27. can

28. couldn

29. couldn't

30. d

31. did

32. didn

33. didn't

34. do

35. does

36. doesn

37. doesn't

38. doing

39. don

40. don't

41. down

42. during

43. each

44. few

45. for

46. from

47. further

48. had

49. hadn

50. hadn't

51. has

52. hasn

53. hasn't

54. have

55. haven

56. haven't

57. having

58. he

59. her

60. here

61. hers

62. herself

63. him

64. himself

65. his

66. how

67. i

68. if

69. in

70. into

71. is

72. isn

73. isn't

74. it

75. it's

76. its

77. itself

78. just

79. ll

80. m

81. ma

82. me

83. mightn

84. mightn't

85. more

86. most

87. mustn

88. mustn't

89. my

90. myself

91. needn

92. needn't

93. no

94. nor

95. not

96. now

97. o

98. of

99. off

100. on

101. once

102. only

103. or

104. other

105. our

106. ours

107. ourselves

108. out

109. over

110. own

111. re

112. s

113. same

114. shan

115. shan't

116. she

117. she's

118. should

119. should've

120. shouldn

121. shouldn't

122. so

123. some

124. such

125. t

126. than

127. that

128. that'll

129. the

130. their

131. theirs

132. them

133. themselves

134. then

135. there

136. these

137. they

138. this

139. those

140. through

141. to

142. too

143. under

144. until

145. up

146. ve

147. very

148. was

149. wasn

150. wasn't

151. we

152. were

153. weren

154. weren't

155. what

156. when

157. where

158. which

159. while

160. who

161. whom

162. why

163. will

164. with

165. won

166. won't

167. wouldn

168. wouldn't

169. y

170. you

171. you'd

172. you'll

173. you're

174. you've

175. your

176. yours

177. yourself

178. yourselves

במהלך העבודה על יצירת האינדקס, חיפשנו באינטרנט רשימה של stop words באנגלית. מצאנו רשימה שנראתה לנו איכותית ורלוונטית, בדקנו את תפקודה בפועל, וראינו שהיא מספקת תוצאות טובות- ולכן בחרנו להשתמש בה.

לצורך תהליך בניית האינדקס, החלטנו ליישם stemming. בחרנו בשיטה זו מאחר והכרנו אותה מהתרגולים הקודמים, והעדפנו לעבוד עם טכניקה מוכרת. לאחר ניסוי ראשוני ראינו שהשיטה מפיקה תוצאות מדויקות ויעילות, ולכן המשכנו להשתמש בה גם בשלבים הבאים של הפרויקט.

חלק שני: התייחסות למשובים וארכיטקטורת המערכת

ביטוי 8 כללי הזהב של שניידרמן במערכת

1. עקביות (**Consistency**)

**הממשק משתמש בעיצוב אחיד של צבעים, תפריטים וכפתורים.**

**לדוגמה: טבלת ה-**Leaderboard **מופיעה תמיד באותו עיצוב ומיקום, תפריט הניווט נמצא תמיד בצד, והצגת הנתונים הגרפיים (גרפים, טבלאות) נשמרת זהה בין המסכים.**

1. קיצורי דרך (**Shortcuts**)

**המעבר בין החלקים העיקריים במערכת מתבצע ישירות דרך תפריט הצד (**Dashboard**, סטטיסטיקות,** Fault Simulator**).**

**במנוע החיפוש (**MQTT**) -תיקון אוטומטי לשגיאות כתיב ("**Did you mean…**") עוזר למצוא מידע מהר גם כשיש טעות בקלט.**

1. משוב למשתמש (**Feedback**)

**בעת הקלדת פרטי התחברות שגויים מוצגת הודעת שגיאה ברורה (**Username or password incorrect**).**

**במנוע החיפוש: מתקבלת הצעה אוטומטית לתיקון שגיאה.**

**בדשבורד: עדכון הניקוד, הצגת תקלות חיות, וכל שינוי נתון מוצגים למשתמש מידית.**

**בסימולטור התקלות, התקלה מוצגת בצבע שתואם את מידת חומרת התקלה (אדום- גבוהה, צהוב- בינונית, ירוק- נמוכה). בעת ביצוע תיקון תקלה, מופעל שעון עצר שמציג את הזמן מתחילת הטיפול בתקלה.**

1. סגירת דיאלוג (**Dialog Closure**)

**לא מתבצע מעבר אוטומטי בין מסכים בסיום פעולה. בסיום סימולציה או פעולה, מוצג ניקוד וסטטוס- והמשתמש מחליט איך להמשיך.**

1. טיפול בשגיאות (**Error Handling**)

**בהתחברות: מופיעה הודעת שגיאה במקרה של שם משתמש/סיסמה לא נכונים.**

**במנוע החיפוש: הודעה על שגיאת כתיב והצעה למילה הנכונה.**

1. אפשרות ביטול (**Easy Reversal of Actions**)

**בטפסים קיימת אפשרות** Clear **(למשל במסך הכניסה), שמאפסת את הקלט.**

1. שליטה פנימית (**Support Internal Locus of Control**)

**כל מעבר בין מסכים, התחלת סימולציה וחיפוש נעשים רק ביוזמת המשתמש.**

**אין פעולות שמתבצעות אוטומטית ללא הפעלה מפורשת.**

1. הפחתת עומס זיכרון (**Reduce Short-Term Memory Load**)

**טבלת ה-**Leaderboard **עם דירוג המשתמש מוצגת תמיד בצד, בכל המסכים. גם אם המשתמש לא בחמישייה הפותחת, מוצגת השורה האישית שלו במקום השישי, יחד עם הדירוג המדויק והניקוד.**

**כל הנתונים הקריטיים מוצגים בזמן אמת. אין צורך לזכור ערכים או תוצאות, הכל מוצג ויזואלית בכל רגע.**

התייחסות למשובים

|  |  |  |
| --- | --- | --- |
| הערת משוב | האם התבצע שינוי באפליקציה בעקבות ההערה? | נימוק |
| **עיצוב** | **כן** | **בוצעו שיפורים בנראות: יישור אלמנטים, הדגשת תפריטים, וממשק עקבי בכל המסכים.** |
| **לשלב יותר צבעים חוץ מכחול** | **כן** | **הוספנו צבעים נוספים לכרטיסי החיישנים.** |
| **להגדיל את הכתב של הסטטיסטיקות/גרפים** | **כן** | **הגדלנו את גודל הפונט של הכותרות והכיתוב על הצירים בגרפים.** |
| **שהגרפים יהיו יותר גדולים** | **כן** | **הגדלנו את שטח הגרפים פי 2.** |
| **הרבה מידע באותו חלון** | **כן** | **חילקנו את המידע לכמה חלונות שונים. בנוסף לכך, הוספנו מדריך למשתמש כדי לחסוך במלל בדפים מסויימים.** |
| **לבחור תאריך מתוך לוח שנה** | **לא** | **במקרה שלנו, בחירת יום, חודש, שנה ושעה באמצעות רשימות נפתחות עדיפה על לוח שנה רגיל כי היא מאפשרת לבחור בקלות טווחי זמן לא שגרתיים. למשל, לבחירת נתונים סטטיסטיים לפי שעה מסוימת, חודש או שנה, בלי להיות מוגבלים לתצוגה יומית בלבד. זה גמיש, מהיר, ומותאם טוב יותר לאופי הסטטיסטיקות במערכת שלנו.** |
| **להשתמש במידע אמיתי ולא פיקטיבי** | **כן** | **השתמשנו במידע פיקטיבי (סימולציה) כי החיישנים לא עבדו באותו רגע, אבל יש לנו חיבור למידע האמיתי, ואפשר להחליף בין מקורות המידע בקלות בעזרת פונקציה ייעודית במערכת.** |
| **לתקן כפתור שלא עובד** | **כן** | **עדכנו את** Fault Simulation **וכעת הכפתור עובד באופן תקין. במהלך המשובים הפיצ'ר עדיין לא היה מוכן עד הסוף, אבל מאז הוא תוקן והושלם.** |
| **תוצאות החיפוש מוצגות בצורת שורשים** (stemmed) | **לא** | **בחרנו להשתמש ב-**stemming **לצורך בניית האינדקס, מכיוון שהוא מהיר ופשוט ליישום, ומאפשר לאחד מילים עם שורש דומה בקלות (למשל:** run ,running, runner **כולם הופכים ל-**run**). השימוש ב-**stem **מסייע להקטין את גודל האינדקס ולשפר את יעילות החיפוש. למרות ש-**lemmatization **מדויק יותר סמנטית, הוא דורש יותר משאבים ותחזוקה.** |
| **אפשר להוסיף מסקנה לנתונים שמוצגים כרגע** | **לא** | **נכון לעכשיו, המערכת מציגה את כל הנתונים בצורה אובייקטיבית, ללא ניתוח בעיקר בגלל פערי מידע מה נחשב לנתון טוב או רע.** |

ציון ה- **SUS** של המערכת

**ציון ה-** SUS **של המערכת שלנו עמד על 88.5 מתוך 100, לפי ממוצע תשובות המשתמשים בשאלון השמישות הסטנדרטי. ציון זה נחשב גבוה מאוד בהשוואה לסטנדרטים הבינלאומיים, כאשר מקובל שציון מעל 68 מעיד על שמישות טובה, ומעל 80 מעיד על שמישות מצוינת. ציון זה הושג בין היתר כיוון שרבים מהמשתמשים סימנו את התשובות הגבוהות ביותר (4 או 5) בשאלון, מה שמעיד שהמערכת נתפסת על ידם כקלה מאוד לשימוש, אינטואיטיבית ומספקת חוויית משתמש מצוינת. המשתמשים תפסו את העבודה עם המערכת כחיובית ומהנה, ולמעשה, ציון כזה משקף רמה גבוהה של שביעות רצון ויעילות בעבודה עם המערכת.**

3 מדדים להצלחת המערכת ע"פ הרצאה 3

**בהתבסס על** 10 המדדים הקריטיים לתעשיית שירותי הענן (**XaaS**) **כפי שמופיעים בשקף 19 של הרצאה 3, בחרנו להסתמך על שלושה מדדים עיקריים להערכת הצלחת המערכת שפיתחנו:**

1. **System Availability** (זמינות מערכת)

**מדד זה בוחן את אחוז הזמן שבו השירות או המערכת פעילים ותקינים. זמינות גבוהה (99.9% ומעלה) מבטיחה שירות רציף למשתמשים.**

**במערכת שלנו, נעשה שימוש בתשתית** Firebase **שנחשבת ליציבה ואמינה.**

**בנוסף, שילבנו מצב סימולציה, שמאפשר שימוש במערכת גם כאשר חיישני הסביבה אינם פעילים בפועל (ע"י הזרמת נתונים פיקטיביים). כך נשמרת זמינות גבוהה לפיצ'רים חיוניים כמו תרגול תיקון תקלות וניווט במערכת גם בתנאי חוסר זמינות של נתונים חיצוניים.**

1. **Response Time** (זמן תגובה)

**מדד זה מתאר את הזמן שעובר מרגע שליחת בקשה למערכת ועד לקבלת תגובה. זמן תגובה קצר הוא מפתח לחוויית משתמש חיובית.**

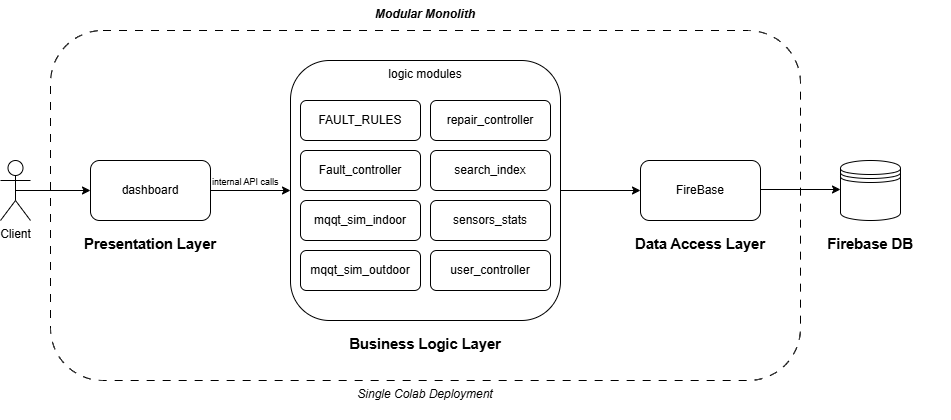
**במערכת שלנו, זמן ההכנה הראשוני (התקנות, טעינת מחברות) אורך כ- 2 דקות בלבד, ולאחר מכן זמן התגובה בין פעולות המשתמש לתגובה מהמערכת הוא מהיר - לרוב שניות ספורות. בכך אנו עומדים בסטנדרטים מקובלים למערכות מבוססות ענן.**

1. **Scalability** (יכולת גידול)

**מדד זה בוחן את היכולת של המערכת לתמוך בהתרחבות- בין אם במספר המשתמשים או בתוספת פיצ'רים.**

**המערכת שלנו בנויה בצורה מודולרית: כל רכיב לוגי (לדוגמה: ניתוח תקלות, שליפת נתונים, טיפול בתיקון) ממומש במחברת נפרדת, וכולן משתלבות בממשק גרפי אחיד. בנוסף, שימוש בתשתית** Firebase **מאפשר הרחבה קלה של בסיס הנתונים, הוספת משתמשים, ותמיכה בפיצ'רים נוספים בעתיד. כך ניתן להמשיך ולפתח את המערכת בקלות גם בקנה מידה רחב יותר.**

ארכיטקטורת המערכת ע"פ הרצאה 7



המערכת שלנו מבוססת על ארכיטקטורת **Modular Monolith**, אשר מפרידה בין מודולים לוגיים אך מריצה את כל הרכיבים באותה סביבת ריצה (Colab). כל רכיב ממומש כמודול עצמאי (מחברת נפרדת) לפי תחום אחריותו, ומחברת אחת (dashboard.ipynb) מרכזת את האינטגרציה, התצוגה והתפעול של כלל המערכת מול המשתמש.

הסבר על מבנה הארכיטקטורה:

* **תיקיית firebase- שכבת גישה לנתונים (Data Access Layer)**

Firebase.ipynb: אחראי על אינטראקציה עם מסד הנתונים בענן (Firebase)- ניהול משתמשים, חיישנים, תקלות וניקוד.

* **תיקיית logic- שכבת הלוגיקה העסקית (Business Logic Layer)**

1. Fault\_controller.ipynb:

מכילה את הלוגיקה לזיהוי תקלות, שליפת תקלות פעילות מהמערכת, ועדכון סטטוס של תקלה קיימת. משמשת את הסימולטור להצגת תקלות חיות ומעקב אחר הטיפול בהן.

2. repair\_controller.ipynb:

אחראית על לוגיקת התיקון: התחלת תהליך תיקון, סימון שלבים כבוצעו, סיום תיקון וחישוב ניקוד בהתאם למהירות התגובה וחומרת התקלה.

3. FAULT\_RULES.ipynb:

כוללת את חוקי הזיהוי והאבחון של תקלות על סמך קריטריונים המוגדרים (כגון סף טמפרטורה, לחות וכו'). מהווה מנוע כללים המייצר תקלות לפי נתוני סנסורים.

4. mqqt\_sim\_indoor.ipynb / mqqt\_sim\_outdoor.ipynb:

שתי מחברות המדמות את זרימת המידע מחיישנים פנימיים וחיצוניים (MQTT). מספקות מקור נתונים חי או מדומה לסנסורים כמו טמפרטורה, לחות ולחץ.

5. sensors\_stats.ipynb:

מטפלת בניתוח סטטיסטי של נתוני סנסורים: שליפה היסטורית לפי תאריך ושעה, חישוב ממוצעים והכנה להצגה גרפית בממשק המשתמש.

6. search\_index.ipynb:

מממשת מנוע חיפוש מבוסס אינדקס: יצירת אינדקס ממסמכי mqtt.org, טיפול ב-stop words ו- stemming, ואחזור קישורים רלוונטיים לפי שאילתא.

7. user\_controller.ipynb:

אחראית על טיפול בנתוני משתמשים ברמה הלוגית: התחברות, שליפת נתונים אישיים, גישה לאלפון, עדכון ניקוד ותפקידים.

* **תיקיית gui- שכבת ההצגה (Presentation Layer)**

dashboard.ipynb: מחברת זו מרכזת את כל המחברות האחרות, בונה את הממשק הגרפי בעזרת Gradio, ומספקת למשתמש גישה נוחה לפיצ'רים כמו ניתוח סטטיסטי, ניהול תקלות, מנוע חיפוש, Leaderboard ועוד.

about.md, user\_guide.md: קבצים סטטיים בפורמט Markdown, המכילים טקסטים קבועים כגון תיאור כללי של המערכת ומדריך למשתמש. השימוש בהם מאפשר הפרדת תוכן תיאורי מהקוד, ומסייע בשמירה על קוד קריא ויעיל בתוך המחברת הראשית.

חלק שלישי: פיצ'ר "סימולטור תקלות"

במסגרת הפרויקט הוספנו סימולטור תקלות אינטראקטיבי, המאפשר למהנדסים לתרגל טיפול בתקלות קיימות.

הפיצ'ר מציג תקלה חיה הנשלפת מהמערכת, כולל תיאור התקלה, חומרתה, פעולות מומלצות לתיקון וממשק לביצוע שלבי התיקון. עם סיום התיקון, מחושב ניקוד מותאם על פי חומרת התקלה ומהירות התגובה.

הפיצ'ר פותח במחברות הלוגיות FAULT\_RULES.ipynb, Fault\_controller.ipynb ו-repair\_controller.ipynb, וכמובן בממשק הראשי (GUI) דרך מחברת ה- Dashboard. הסימולציה מוצגת למנהל בממשק ייעודי הכולל תצוגת תקלה חיה, סטטוס תיקון וניקוד.

תוספת זו מיישמת את מרכיב הגיימיפיקציה הנדרש בהגדרות הפרויקט, ומעודדת שיפור תהליכי תחזוקה, מעקב ביצועים ותחרותיות חיובית בין המהנדסים.