

# מבחן: פריסת אפליקציית Streamlit ל-Minikube

## ה Disk On Key מרפיח

יוסף מ"מ ערבה מצא במילואים האחרונים ברפיח disk on key, לאחר שובו מהמילואים, יוסף בדק את הכונן (כמובן שמסיבות הגנה הוא השתמש במחשב ישן, חסר ערך, ללא מידע שלא מחובר לרשת) במקום לגלות מפתחות לביטקוין (כפי שהוא קיווה), התברר שהכונן מכיל מספר קבצי טקסט, ללא שמות אינפורמטיביים, המכילים נתונים טבלאיים.

באחד הקבצים יוסף הצליח לזהות אוסף קואורדינטות של מיקומים גיאוגרפיים (קווי אורך וגובה) ולחרדתו התברר שמדובר באוסף נקודות שמרכזן הוא בניין נמרודי.

יוסף האחראי, העביר את הכונן הנייד לידי הגורם המודיעיני הכי בכיר שהוא מכיר - מני. מני לאחר בירורים בתוך היחידה, החליט להעביר את הנתונים להמשך ניתוח לצוות אנליסטים גאונים - אלו שיודעים לזהות דפוסים בתוך סדרות של מספרים ונתונים.

כאיש פיתוח, מני בנה באופן זריז דאשבורד ב-streamlit המציג את הנתונים הגולמיים, ויזואליזציה שלהם, ויזואליזציה של הקואורדינטות על גבי מפה.

תפקידכם לבנות ולהגדיר תהליך פריסה, שיועבר לצוות תשתיות של הענן הסודי של היחידה. במסגרת התהליך עליכם להגדיר קובץ דוקר עבור האפליקציה, מניפסטים עבור קוברניטיס וקובץ המכיל את הפקודות CLI שהשתמשתם בהם, כל התכולות צריכות לעלות לגיטהאב באופן מסודר ובחלוקה לבראנצ'ים. כך שנוכל להעביר את כל התהליך באופן מסודר ומתועד לצוות תפעול ותשתיות

במבחן זה תבחנו על פריסת אפליקציית (Python Streamlit) על גבי Kubernetes באמצעות Minikube. התרגיל משלב את הידע שלכם ב-Docker עם העולם של Kubernetes. האפליקציה הנדרשת יושבת בגיטהאב (לינק בהמשך)

## חומר עזר מותר:

## אין להשתמש בחומר שניתן בכיתה או קבצים אישיים.

<https://minikube.sigs.k8s.io/docs/handbook/>

<https://octopus.com/devops/kubernetes-deployments/kubernetes-yaml-types/>

<https://medium.com/geekculture/cheatsheet-for-kubernetes-minikube-kubectl-5500ffd2f0d5>

## דרישות מוקדמות

לפני שמתחילים, תוודאו שמותקנים אצלכם:

- Docker Desktop
- Minikube
- kubectl
- חשבון ב-Docker Hub (איש)
- Git
- חשבון אישי ב-GitHub

## מה להגיש?

צרו פרויקט גיט עם הקבצים הבאים:

1. **Dockerfile** - קובץ ה-Docker שכתבתם
2. **pod.yaml** - קובץ ה-Pod ל-Kubernetes
3. **service.yaml** - קובץ ה-Service ל-Kubernetes
4. **commands.txt** - קובץ טקסט עם כל פקודות ה-CLI שהרצתם (מהשלב הראשון ועד האחרון)
5. **screenshot.png** - צילום מסך של האפליקציה פועלת בדפדפן (כולל ה-URL בשורת הכתובת)

**שימו לב מצופה שכל קובץ יהיה בבראנץ נפרד עם קובץ MD הכולל הסבר על מהות ומטרת הקובץ ורק לבסוף יש לאחד את הכל ל main.**  
**יש להעלות את הפרויקט לגיטהאב ולהדביק את הקישור למטלת ההגשה במודל.**

## שלב 1: הכנת הסביבה

### 1.1 הורדת הקוד מ-GitHub

זוהי האפליקציה שעליכם לפרוס - הורידו את הקוד מגיטהאב.  
לא לשכוח לשמור את פקודות ה-CLI הרלוונטיות להורדת קבצים מ-github ב-commands.txt

[https://github.com/yosefanaliza/streamlit\\_minikube\\_app](https://github.com/yosefanaliza/streamlit_minikube_app)

### 1.2 הפעלת Minikube

לא לשכוח לשמור את פקודות ה-CLI הרלוונטיות להפעלת מיניקיוב ובדיקת סטטוס ראשונית ב-commands.txt

## שלב 2: בניית Docker Image (ניקוד 27%)

### 2.1 כתיבת Dockerfile עבור האפליקציה

בגייט יש להשתמש בבראנץ נפרד הכולל את הקובץ דוקר וקובץ הסבר פשוט לגביו.  
לא לשכוח לשמור את פקודות ה CLI הרלוונטיות לגייט ב `commands.txt`.

### 2.2 בניית ה-Image

לא לשכוח לשמור את פקודות ה CLI הרלוונטיות לבניית התמונה בדוקר ב `commands.txt`.

### 2.3 בדיקה מקומית

יש להריץ את הקונטיינר בעזרת דוקר ולבדוק שאכן ניתן להתחבר אליו ולראות את הדאשבורד בדפדפן.  
לא לשכוח לשמור את פקודות ה CLI להרצת קונטיינר בדוקר וסטטוס ב `commands.txt`.

### 2.4 העלאה ל-Docker Hub

לא לשכוח לשמור את פקודות ה CLI הרלוונטיות לגישה לדוקר האב - תיוג, דחיפה ו-וילידציה ב-`commands.txt`.

## שלב 3: פריסה ל-Kubernetes (ניקוד 53%)

### 3.1 כתיבת קובץ `pod.yaml`

בגייט יש להשתמש בבראנץ נפרד הכולל את הקובץ `pod.yaml` וקובץ הסבר פשוט לגביו.  
לא לשכוח לשמור את פקודות ה CLI הרלוונטיות לגייט ב `commands.txt`.

### 3.2 כתיבת קובץ `service.yaml`

בגייט יש להשתמש בבראנץ נפרד הכולל את הקובץ `service.yaml` וקובץ הסבר פשוט לגביו.  
לא לשכוח לשמור את פקודות ה CLI הרלוונטיות לגייט ב `commands.txt`.

## שלב 4: הרצת האפליקציה ב-Minikube (ניקוד 15%)

### 4.1 יישום (apply) הקבצים

לא לשכוח לשמור את פקודות ה CLI הרלוונטיות ליישום הקבצים ב `commands.txt`.

### 4.2 בדיקת סטטוס

לא לשכוח לשמור את פקודות ה CLI הרלוונטיות לבדיקת סטטוס ב `commands.txt`.

### 4.3 גישה לאפליקציה

`screenshot.png` - צילום מסך של האפליקציה פועלת בדפדפן (כולל ה-URL בשורת הכתובת)  
בגייט יש להשתמש בבראנץ נפרד הכולל את הקובץ `screenshot.png` וקובץ הסבר פשוט לגביו.  
לא לשכוח לשמור את פקודות ה CLI הרלוונטיות להרצת האפליקציה ולגייט ב `commands.txt`.

## שלב 5: בדיקות נוספות (ניקוד 5%)

### 5.1 צפייה בלוגים

לא לשכוח לשמור את פקודות הרלוונטיות לבדיקת לוגים ה CLI ב `commands.txt`.

### 5.2 מידע מפורט על ה-Pod

לא לשכוח לשמור את פקודות ה CLI לקבלת מידע מפורט על ה POD ב `commands.txt`.

---

## טיפים חשובים

- שימו לב לשם המשתמש ב-Docker Hub - תוודאו שהוא תואם בכל מקום
- בדקו `syntax` - טעויות קטנות ב-YAML יכולות לגרום לכשלים
- השתמשו ב-`kubectl describe` - זה עוזר לזהות בעיות
- קראו הודעות שגיאה - Kubernetes נותן הודעות שגיאה מפורטות
- אל תשכחו את `--server.address=0.0.0.0` - בלעדיו Streamlit לא יהיה נגיש מחוץ ל-container (המושג נקרא bind address והוא מציין מאיזה IP ניתן לגשת לשרת ו 0.0.0.0 מאפשר גישה מכל IP)

פקודת ההרצה עם הקונפיגורציה הנ"ל:

```
streamlit run your_app.py --server.port 8080 --server.address=0.0.0.0
```

## שאלות נפוצות

ש: ה-Pod תקוע ב-`ImagePullBackOff`

ת: בדקו ש-image name נכון ושהעליתם אותו ל-Docker Hub. נסו `kubectl describe pod` לפרטים.

ש: לא מצליח להיכנס לאפליקציה

ת: ודאו שה-Service מסוג NodePort והשתמשו ב-`minikube service streamlit-service` `--url` לקבלת ה-URL הנכון.

ש: האפליקציה לא מציגה נתונים

ת: בדקו את הלוגים של ה-Pod עם `<pod-name> kubectl logs`.

בהצלחה!