1. **AirFlow תהליך לילי** - שמוריד את הקבצי CSV למכונה ועושה להם UnZip:
   1. מוחק את התיקיות מאתמול ויוצר חדשות.
   2. מוריד את הקבצים.
   3. עושה unzip .
   4. מסדר את הקבצים הרלוונטים בתתיקייה אחת ואת השאר מוחק.
   5. **בPandas** הופך את כל הקבצי הCSV ל DataFrame.
   6. מסנן את כל הנתונים לגוש דן בלבד לפי קובץ Zone.
   7. יוצר DataFrame אחד שמאחד את כל הנתונים על כל Trip.
   8. שומר את הנתונים **לMongo** כJson.
2. **AirFlow Generator** - מתחבר ל API של משרד התחבורה ומביאי נתונים בזמן אמת:
   1. שולף מהMongo את המספר קווים הקיימים ואת הTrip שלהם.
   2. כל 15 שניות:
      1. מתחבר לAPI ומחפש מידע על המיקומי של כל אוטובוסים
      2. סינון שמביא את המידע רק של הקווים הנבחרים.
      3. שולח את הנתונים לkinesis.
3. **Kinesis** – צינור מעבר מידע ל spark
4. **Spark** – עיבוד המידע.
   1. שליפת הנתונים מהmongo,
   2. ע"י spark
      1. איחוד של הנתונים מהmongo עם הנתונים שמגיעים מהKinesis
      2. סידור השעות
      3. מציאת התחנה הכי קרובה למיקום של האוטובוס
      4. העשרת המידע של בSpark של מספר התחנה הכי קרובה + המרחק של האוטובוס מהתחנה והוספת השעה שהאוטובוס מיועד להגיע לתחנה ע"י התכנון.
      5. מעבירים את המידע לkinesis.
5. **Kinesis** – צינור מעבר מידע 2:
   1. שומרים את כל הנתונים שמתקבלים בArchive ב S3 כקבצי parquetע"י **kinesis firehose**
      1. שומר בpartitions לפי שעה.
   2. **kinesis data analytics** – סינון המידע בStream ביצירת Topics חדשים
      1. topic חדש שבודק אם האוטובוס נמצא פחות מ50 מטר מהתחנה הקרובה  
         שומר את המידע מתי האוטובוס אגיע לתחנה מול השעה המתוכננת.

שומר בpartition לפי מספר קו ב S3 כקבצי parquetע"י **kinesis firehose**

1. **Lambda** – מקבל את המיקומים של האוטובוסים כשהם בתחנה ושומר את הנתונים בMongo
2. **aws glue** - מבצע crawler כל שעה בשביל לדעכן את הpartition החדשים שנוצרים.
3. **Athena** - מפיקה דוחות קבצים בS3 לבדוק זמן הגעה בפועל עבור כל קו וסטטיסטיקות עבור כל מפעיל.
4. **Tableau** – מציג מידע:
   1. מידע של הדוחות.
   2. מפה של כל המסלולים ואיפה נמצא כל אוטובוס בזמן אמת מתוך הHBase.