

Οικονομικό Πανεπιστήμιο Αθηνών

Τμήμα Πληροφορικής

ΠΜΣ ΣΤΗΝ ΑΝΑΠΤΥΞΗ & ΑΣΦΑΛΕΙΑ ΠΛΗΡΟΦΟΡΙΑΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

Εαρινό Εξάμηνο 2024-2025

Τεχνολογίες Ψηφιακών Υποδομών

Σύστημα Επεξεργασίας Ροών Βίντεο Κυκλοφορίας

Δεύτερη Φάση Εργασίας

Διδάσκων: Β. Καλογεράκη

Βοηθός: Μιχαήλ Τσένος

Ευγένιος Γκρίτσης, f3312306

Ελένη Νικολάου, f3312405

Αικατερίνη Πετρούλια, f3312411

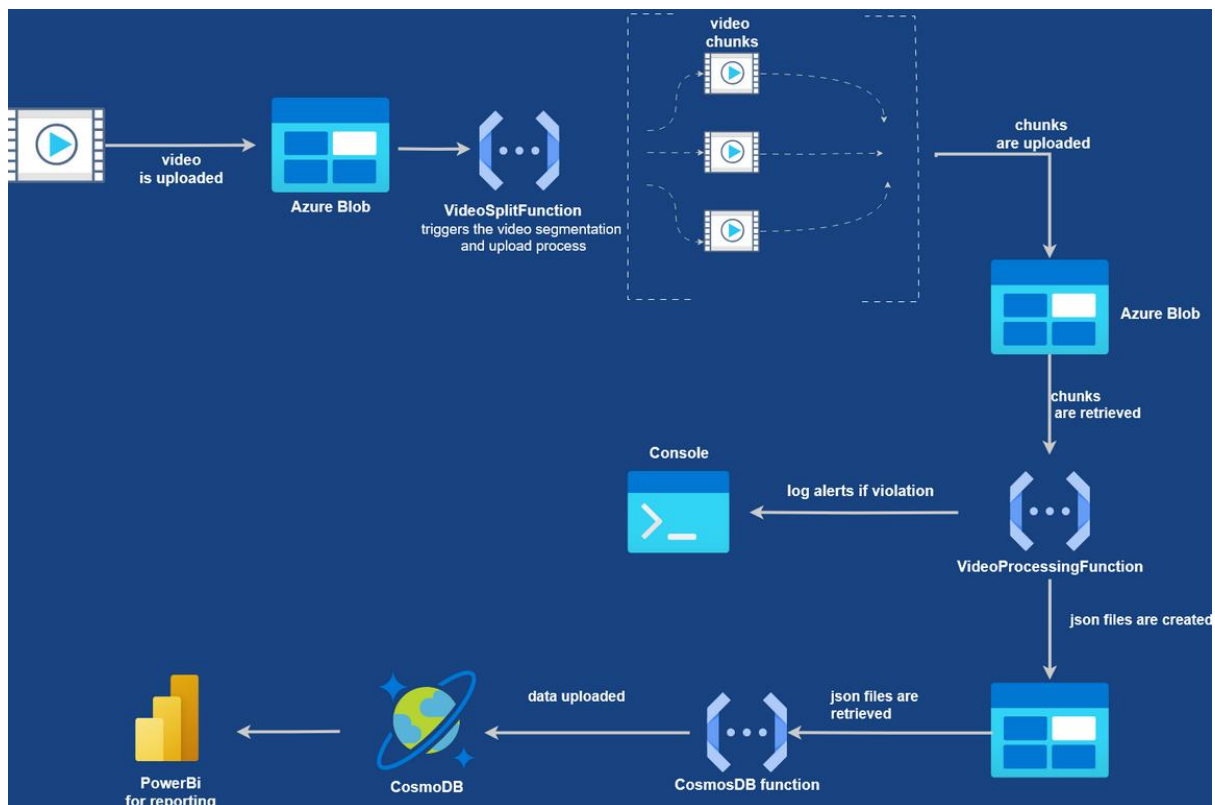
Νικολέττα Τερζίδου, f3312412

Github: <https://github.com/eGkritis/SmartCity-Traffic-Processing>

Περιεχόμενα

Αρχιτεκτονική	3
Περιγραφή Κώδικα.....	5
Εφαρμογή.....	10
Αναφορά PowerBI	19
Εργαλεία	20

Αρχιτεκτονική



Το Σύστημα Επεξεργασίας Ροών Βίντεο Κυκλοφορίας βασίζεται σε μία cloud-native αρχιτεκτονική με στόχο την αποδοτική επεξεργασία, ανάλυση και αξιοποίηση δεδομένων από κάμερες διαχείρισης κυκλοφορίας. Η ροή επεξεργασίας αποτελείται από τα εξής στάδια:

Αρχική Αποθήκευση του Βίντεο:

Το αρχικό βίντεο αποθηκεύεται σε Azure Blob Storage (raw-video container).

Διάσπαση του Βίντεο:

Μέσω μίας Azure Function που ονομάζουμε VideoSplitFunction το βίντεο διασπάται σε clip δύο λεπτών. Κατόπιν αποθηκεύεται σε νέο container του Azure Blob Storage (splitted-videos).

Επεξεργασία των clips:

Τα clips ανακτώνται από το Blob, και επεξεργάζονται από νέα Azure Function, που ονομάζουμε VideoProcessingFunction. Η συνάρτηση αυτή χρησιμοποιεί την βιβλιοθήκη `opencv` και το YOLO μόντελο για την ανίχνευση οχημάτων σε κάθε καρέ του βίντεο. Υπολογίζει ταχύτητες και κατεύθυνση για κάθε όχημα με βάση τη μετατόπισή του σε διαδοχικά καρέ.

Σε περίπτωση που υπάρχουν παραβιάσεις των ορίων ταχύτητας καταγράφει προειδοποιήσεις σε πραγματικό χρόνο. Συλλέγει και αποθηκεύει τα στοιχεία κάθε οχήματος (μέγιστη ταχύτητα, κατεύθυνση, θέση, χρονική σήμανση) σε νέο container του Azure Blob (`processed-stats`).

Αποθήκευση Δεδομένων στη Βάση:

Νέα Azure Function, που λέγεται CosmosDB Function, αναλαμβάνει να αποθηκεύσει τα επεξεργασμένα δεδομένα στο Azure Cosmos Database σε κατάλληλο json format.

Αναφορές:

Στη συνέχεια, παράγονται αναφορές και γίνεται οπτικοποίηση των δεδομένων με PowerBI. Τα δεδομένα αναλύονται και εξάγονται στατιστικά και στοιχεία που αφορούν:

- Την ταχύτητα με την οποία κινείται κάθε όχημα.
- Τον αριθμό των οχημάτων ανά ρεύμα.
- Τον αριθμό των οχημάτων που έχουν υπερβεί το όριο ταχύτητας, με όρια: 90 χλμ/ώρα για αυτοκίνητα, 80 χλμ/ώρα για φορτηγά.
- Τον αριθμό των οχημάτων ανά ρεύμα και ανά 5λεπτο.
- Τη μέση ταχύτητα ανά ρεύμα και ανά 5λεπτο, καταγεγραμμένη σε αρχείο log, π.χ.: (inbound, 1st 5min, 60km/h), (outbound, 5th 5min, 72km/h)

Η αρχιτεκτονική συνδυάζει real-time ειδοποιήσεις και ανάλυση, προσφέροντας μία λύση διαχείρισης κυκλοφορίας με έμφαση στην αποδοτικότητα, την κλιμακωσιμότητα και την αξιοπιστία.

Περιγραφή Κώδικα

Η **HTTP-VIDEO-SPLIT-FUNCTION** αποτελεί το αρχικό στάδιο στην επεξεργασία ροών βίντεο και έχει ως στόχο τον κατακερματισμό ενός μεγάλου αρχείου σε μικρότερα, διαχειρίσιμα segments.

Πρόκειται για μια Azure Function που ενεργοποιείται μέσω **HTTP** αιτήματος και λαμβάνει ως είσοδο ένα βίντεο (**.mp4**) από το **Azure Blob Storage (raw-video container)**. Το βίντεο αποθηκεύεται προσωρινά στο τοπικό σύστημα και στη συνέχεια, με τη χρήση του εργαλείου **FFmpeg**, διαχωρίζεται σε αποσπάσματα διάρκειας δύο λεπτών. Τα παραγόμενα **segments** αποθηκεύονται ξανά σε ξεχωριστό container του Azure Blob Storage (**splitted-videos**), ώστε να είναι έτοιμα για περαιτέρω επεξεργασία.

```
# Configuration
input_blob_name = "test.mp4"
input_container = "raw-video"
output_container = "splitted-videos"
segment_time = "120" # 2 minutes in seconds

temp_dir = "/tmp"
input_video = f"{temp_dir}/{input_blob_name}"
output_prefix = f"{temp_dir}/split_"
```

Η συνάρτηση **υποστηρίζει fallback** μηχανισμούς για την εύρεση του ffmpeg (διασφαλίζοντας διαθεσιμότητα, αξιοπιστία), ενώ παράλληλα ενσωματώνει μηχανισμό καταγραφής συμβάντων (**logging**), που διευκολύνει τον εντοπισμό προβλημάτων και την παρακολούθηση της διαδικασίας. Σε περίπτωση αποτυχίας οποιουδήποτε βήματος, η λειτουργία επιστρέφει κατάλληλο **μήνυμα σφάλματος** μέσω HTTP.

Η συγκεκριμένη υλοποίηση επιτρέπει την οριζόντια **κλιμάκωση**, καθώς κάθε instance της συνάρτησης μπορεί να εκτελεί ανεξάρτητα το διαχωρισμό διαφορετικών βίντεο, συμβάλλοντας στην ταχύτερη επεξεργασία μεγάλου όγκου δεδομένων.

HTTPVIDEOPROCESSINGFUNCTION

Αποτελείται από δύο κύρια στοιχεία: Το `processing.py` περιέχει τον βασικό κώδικα για την ανίχνευση και παρακολούθηση οχημάτων στα βίντεο. Το `__init__.py` λειτουργεί ως ενδιάμεσος, καθιστώντας τις βασικές συναρτήσεις του `processing.py` άμεσα διαθέσιμες για χρήση σε άλλα μέρη της εφαρμογής.

`__INIT__.PY`

Η λειτουργία της ενεργοποιείται μέσω **HTTP αιτήματος** κάνει την επεξεργασία όλων **των αρχείων τύπου .mp4** που βρίσκονται αποθηκευμένα σε καθορισμένο container (**splitted-videos**) του Azure Blob Storage. Για κάθε διαθέσιμο βίντεο, η συνάρτηση το

κατεβάζει προσωρινά σε tempfile, το περνά από μια διαδικασία ανίχνευσης και παρακολούθησης οχημάτων (μέσω της συνάρτησης **process_video_clip**) και στη συνέχεια εξάγει στατιστικά στοιχεία.

Η ανάλυση επικεντρώνεται μόνο σε

δύο τύπους οχημάτων – αυτοκίνητα και φορτηγά – αγνοώντας άλλα είδη όπως μηχανές ή ποδήλατα. Για κάθε όχημα, κρατιέται μόνο η **μέγιστη ταχύτητα** που καταγράφηκε κατά τη διάρκεια του βίντεο, ώστε να αποφεύγεται η διπλή καταμέτρηση τους. Τα **στατιστικά συνοψίζονται** σε ένα αρχείο **JSON** το οποίο περιλαμβάνει πληροφορίες για το όνομα του αρχείου, την ώρα επεξεργασίας, τον συνολικό αριθμό διαφορετικών οχημάτων και ιδιότητες του βίντεο (όπως διάρκεια ή ανάλυση, εφόσον είναι διαθέσιμα). Τα αποτελέσματα ανεβαίνουν σε ξεχωριστό container (**processed-stats**) στο Azure Blob Storage.

```
# Configuration
INPUT_CONTAINER = "splitted-videos"
OUTPUT_CONTAINER = "processed-stats"
AZURE_STORAGE_CONNECTION_STRING = os.getenv("AzureWebJobsStorage")

# Speed limits (km/h)
SPEED_LIMITS = {
    "car": 90,
    "truck": 80,
    "emergency": 130
}
```

Επιπλέον, η συνάρτηση είναι **ανθεκτική** σε αποτυχίες: αν κάποιο αρχείο δεν μπορέσει να επεξεργαστεί, το καταγράφει σε ξεχωριστή λίστα και συνεχίζει με τα υπόλοιπα, διασφαλίζοντας ότι η συνολική λειτουργία δεν διακόπτεται από επιμέρους σφάλματα.

```
# Prepare response
response = {
    "status": "completed",
    "processed_files": processed_files,
    "failed_files": failed_files,
    "processed_count": len(processed_files),
    "failed_count": len(failed_files),
    "completion_time": datetime.datetime.now().isoformat()
}
```

Τέλος, η συνάρτηση παράγει **αναφορά** για το ποια αρχεία επεξεργάστηκαν επιτυχώς, ποια απέτυχαν, καθώς και τον συνολικό αριθμό τους, προσφέροντας ευκολία στην παρακολούθηση της διαδικασίας.

Το **PROCESSING.PY** χρησιμοποιεί το μοντέλο YOLOv3-tiny για την ανίχνευση οχημάτων (αυτοκίνητα και φορτηγά) σε κάθε καρέ του βίντεο και μια ειδικά σχεδιασμένη κλάση **VehicleTracker** για να παρακολουθεί την πορεία και ταχύτητα κάθε οχήματος με βάση τη θέση του σε διαδοχικά καρέ.

```
# ----- Constants -----
DASHED_LINE_DISTANCE = 20          # meters between dashed lines
MIN_SPEED_THRESHOLD = 5            # km/h - minimum speed to consider
MAX_SPEED_THRESHOLD = 200         # km/h - maximum plausible speed
PIXELS_DASHED_LINE = 245
# pixel to meter conversion factor
PIXELS_PER_METER = PIXELS_DASHED_LINE / DASHED_LINE_DISTANCE
SPEED_SMOOTHING_WINDOW = 3        # number of frames to average speed over
CONFIDENCE = 0.6                  # confidence threshold for car/truck detection
```

Ο tracker μπορεί να αντιστοιχίζει νέες ανιχνεύσεις σε σχέση με τις προηγούμενες με βάση την εγγύτητα και το χρονικό διάστημα και να

υπολογίζει ταχύτητα με βάση την απόσταση που διανύθηκε και τον χρόνο που πέρασε.

```
if 'speed' in v and len(v['positions']) >= 3:
    dx = v['positions'][-1][0] - v['positions'][0][0]
    dy = v['positions'][-1][1] - v['positions'][0][1]
    direction = (
        'inbound'
        if (abs(dx) > abs(dy) and dx < 0) or (abs(dy) >= abs(dx) and dy < 0)
        else 'outbound'
    )
```

Επίσης, κρατά τις τελευταίες μετρήσεις ταχύτητας κάθε οχήματος και υπολογίζει τον μέσο όρο τους. Έτσι αποφεύγονται απότομα скаμπανεβάσματα, και η τελική ταχύτητα που εμφανίζεται είναι πιο ομαλή και αξιόπιστη. Εάν κάποιο όχημα υπερβεί τα 130 km/h, τότε παράγεται προειδοποιητικό μήνυμα (alert) στο log. Επίσης, προσδιορίζεται η κατεύθυνση κίνησης (**inbound/outbound**), ανάλογα με την κατεύθυνση της πορείας του στο βίντεο.

```
class VehicleTracker:
    def __init__(self):
        """Initialize vehicle tracker with empty structures"""
        self.tracked_vehicles = {} # Active vehicles being tracked
        self.available_ids = set() # Reusable vehicle IDs
        self.next_id = 1 # Next available new ID
        self.alerted_vehicles = {} # Track last alert time per vehicle
        self.max_speeds = {} # Track maximum speed per vehicle
```

Η συνάρτηση **process_video_clip** φορτώνει το YOLO μοντέλο, διαβάζει το βίντεο frame-by-frame και κάνει ενημέρωση του VehicleTracker, ενώ συλλέγει τις πληροφορίες των οχημάτων και τις συνοψίζει, κρατώντας μόνο την εγγραφή με τη μέγιστη ταχύτητα για κάθε όχημα. Τέλος, δημιουργεί ένα λεξικό αποτελεσμάτων που περιλαμβάνει στοιχεία του βίντεο· το πλήθος και τις ιδιότητες των οχημάτων, δηλαδή ταχύτητα, κατεύθυνση, θέση και timestamp.

```
# Filter to only keep max speed per vehicle
unique_vehicles = {}
for vehicle in results["vehicles"]:
    vid = vehicle['id']
    if vid not in unique_vehicles or vehicle['speed'] > unique_vehicles[vid]['speed']:
        unique_vehicles[vid] = vehicle
```

HTTPCosmosDB

Η Azure Function αυτή ενεργοποιείται μέσω HTTP αιτήματος και διαβάζει τα αποτελέσματα από το container "**processed-stats**" του Azure Blob Storage. Για κάθε αρχείο JSON, το κατεβάζει προσωρινά (τοπικά στη μνήμη), και διαβάζει και επεξεργάζεται το περιεχόμενό του.

Αναγνωρίζει και επεξεργάζεται τα δεδομένα κάθε οχήματος, προσθέτοντας μοναδικό αναγνωριστικό. Καθώς ο VehicleTracker επεξεργάζεται κάθε clip ανεξάρτητα, δίνει στα

οχήματα τοπικά ids (π.χ. 0, 1, 2), τα οποία μπορεί να επαναλαμβάνονται σε διαφορετικά clips. Για να αποφευχθούν επαναλήψεις, εδώ η Azure Function δίνει τελικά μοναδικό ID που προκύπτει από τον συνδυασμό του αριθμού του clip και του τοπικού id του οχήματος (π.χ. στο clip 3, το πρώτο όχημα λαμβάνει id '3_1').

Επίσης, προσαρμόζει τα timestamps των οχημάτων ώστε να λαμβάνουν υπόψη τη συνολική διάρκεια των προηγούμενων clips, προσθέτοντας ένα χρονικό offset. Έτσι τα timestamps των οχημάτων έχουν κατάλληλη αντιστοίχιση στο αρχικό πλήρες βίντεο.

Η συνάρτηση οργανώνει όλα τα στοιχεία σε ένα ενιαίο JSON αρχείο το οποίο αποθηκεύεται σταδιακά σε μια βάση δεδομένων **Cosmos DB**. Το έγγραφο περιλαμβάνει το όνομα του βίντεο, το συνολικό αριθμό οχημάτων, λεπτομέρειες για κάθε όχημα (όπως ταχύτητα, θέση, κατεύθυνση, timestamp), και μεταδεδομένα για το βίντεο (π.χ. διάρκεια, ανάλυση).

```
# Prepare Cosmos DB document
document = {
    "id": json_data["video_name"].replace(".mp4", ""),
    "video_name": json_data["video_name"],
    "partitionKey": json_data["video_name"],
    "processing_time": json_data["processing_time"],
    "total_vehicles": json_data["total_vehicles"],
    "vehicles": json_data["vehicles"], # List of vehicle dicts
    "video_metadata": json_data["video_metadata"]
}
```

Καθ' όλη τη διάρκεια της διαδικασίας, η λειτουργία διαχειρίζεται σφάλματα που μπορεί να προκύψουν από σφάλματα στα JSON αρχεία, προβλήματα σύνδεσης ή χειρισμού της Cosmos DB και της Azure Blob Storage, καταγράφοντας τα αντίστοιχα λάθη χωρίς να διακόπτει τη συνολική ροή.

```
except KeyError as e:
    logger.error(f"Missing environment variable: {str(e)}")
    return func.HttpResponse(
        f"Configuration error: Missing environment variable {str(e)}",
        status_code=500
    )
```

Τέλος, επιστρέφει μια αναφορά (JSON απόκριση) με τον αριθμό των αρχείων που επεξεργάστηκαν επιτυχώς και των σφαλμάτων, διευκολύνοντας την παρακολούθηση και τον έλεγχο της λειτουργίας.

CosmosDB-QUERIES

Για την υλοποίηση των ερωτημάτων κατασκευάστηκαν τα εξής DB queries:

```
-- Query 1
SELECT vehicle.id, vehicle.speed
FROM doc
JOIN vehicle IN doc.vehicles
WHERE vehicle.type = 'car'

-- Query 2
SELECT vehicle.direction, COUNT(1) AS vehicle_count
FROM doc
JOIN vehicle IN doc.vehicles
GROUP BY vehicle.direction

-- Query 3
SELECT VALUE COUNT(1)
FROM doc
JOIN vehicle IN doc.vehicles
WHERE (vehicle.type = 'car' AND vehicle.speed > 90)
OR (vehicle.type = 'truck' AND vehicle.speed > 80)

-- Query 4
SELECT
    vehicle.direction,
    FLOOR(vehicle.timestamp / 300) AS interval_5min,
    COUNT(1) AS vehicle_count
FROM doc
JOIN vehicle IN doc.vehicles
GROUP BY
    vehicle.direction,
    FLOOR(vehicle.timestamp / 300)

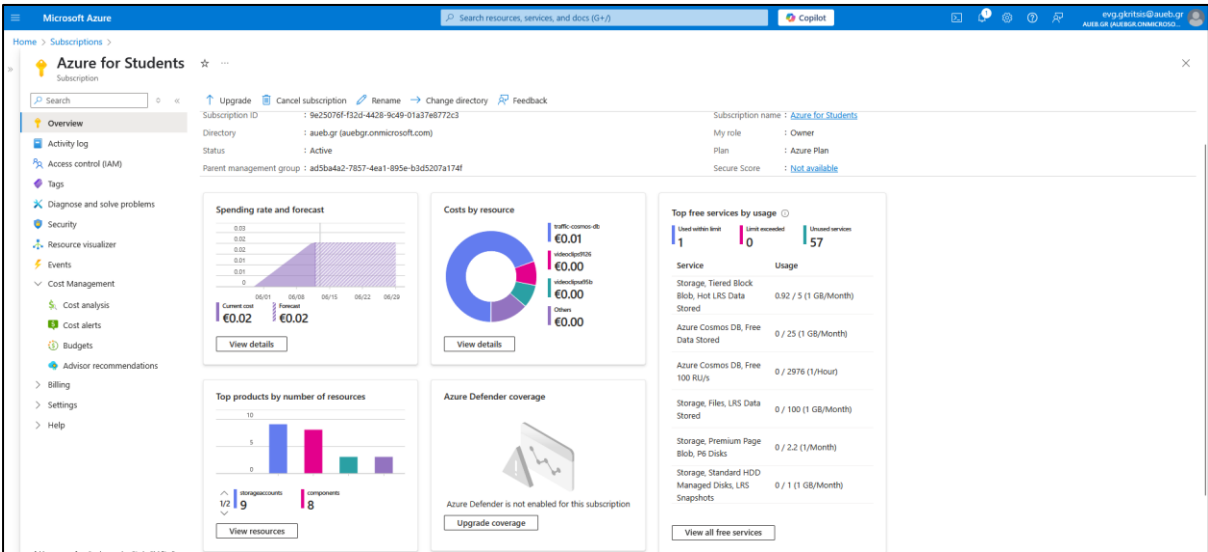
-- Query 5
SELECT
    v.direction,
    FLOOR(v.timestamp / 300) AS interval_5min,
    AVG(v.speed) AS avg_speed
FROM c
JOIN v IN c.vehicles
WHERE IS_DEFINED(v.speed) AND IS_DEFINED(v.direction) AND IS_DEFINED(v.timestamp)
GROUP BY
    v.direction,
    FLOOR(v.timestamp / 300)
```

Με τα queries αυτά εντοπίζουμε την ταχύτητα συγκεκριμένων οχημάτων, υπολογίζουμε πόσα οχήματα κινούνται προς κάθε κατεύθυνση, πόσα υπερβαίνουν προκαθορισμένα όρια ταχύτητας ανάλογα με τον τύπο τους, και πώς κατανέμονται τα οχήματα και οι μέσες ταχύτητές τους σε διαστήματα των 5 λεπτών.

Τα αποτελέσματα των queries χρησιμοποιούνται για την εξαγωγή στατιστικών συμπερασμάτων που αναλύονται με το PowerBI.

Εφαρμογή

Για την εφαρμογή του συστήματος χρησιμοποιήθηκε λογαριασμός Azure for students.



Παρακάτω φαίνονται τα Containers που χρησιμοποιήσαμε.

The screenshot shows the 'trafficvideo | Containers' page. The table lists the following containers:

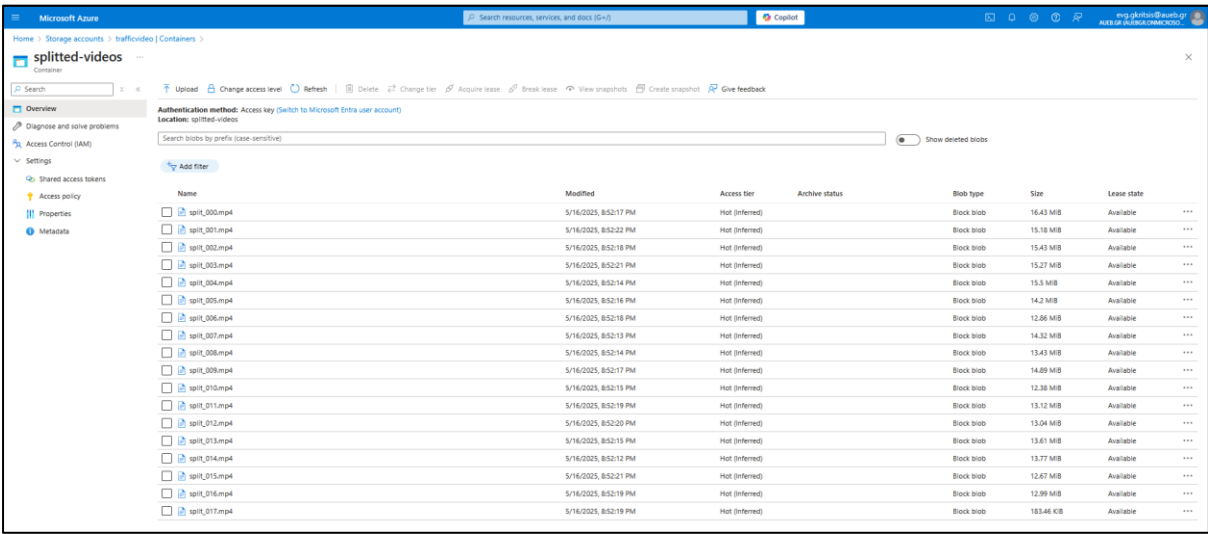
Name	Last modified	Anonymous access level	Lease state
logs	5/14/2025, 1:40:41 PM	Private	Available
azure-webjobs-hosts	5/14/2025, 4:17:22 PM	Private	Available
azure-webjobs-secrets	5/14/2025, 5:42:37 PM	Private	Available
processed-stats	5/15/2025, 3:06:53 PM	Private	Available
raw-video	5/14/2025, 6:21:05 PM	Private	Available
scm-releases	5/14/2025, 5:20:08 PM	Private	Available
splitted-videos	5/14/2025, 6:30:37 PM	Private	Available

Το αρχικό βίντεο αποθηκεύτηκε στο raw-video container.

The screenshot shows the 'raw-video' container details page. The authentication method is 'Access key (Switch to Microsoft Entra user account)' and the location is 'raw-video'. The table lists the following blobs:

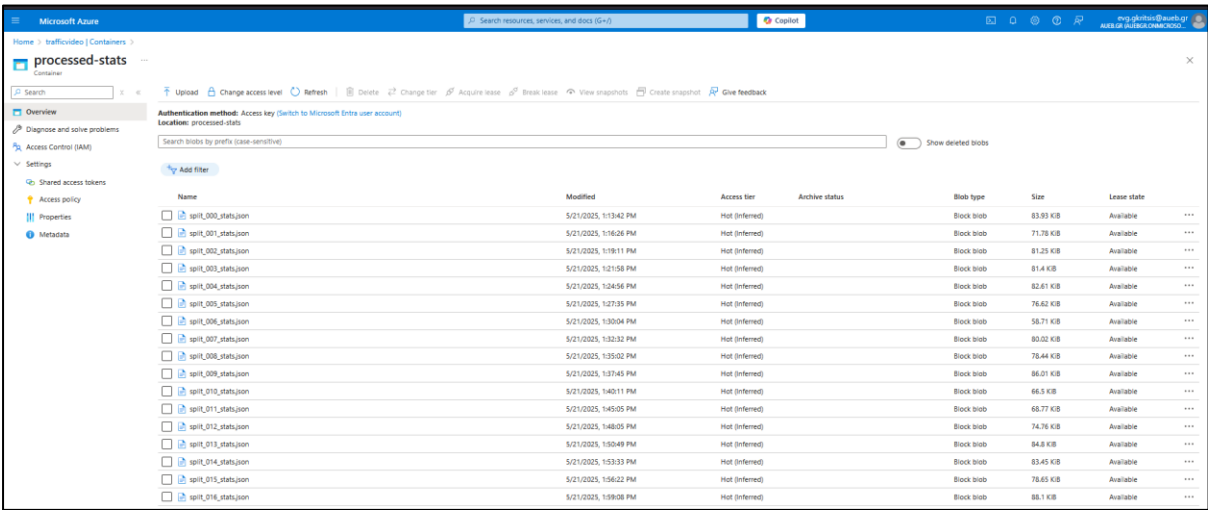
Name	Modified	Access tier	Archive status	Blob type	Size	Lease state
test.mp4	5/14/2025, 4:38:05 PM	Hot (Inferred)		Block blob	239.23 MiB	Available

Παρήχθησαν τα εξής clips από την VideoSplittingFunction:



Name	Modified	Access tier	Archive status	Blob type	Size	Lease state
split_000.mp4	5/16/2025, 8:52:17 PM	Hot (Inferred)		Block blob	16.43 MiB	Available
split_001.mp4	5/16/2025, 8:52:22 PM	Hot (Inferred)		Block blob	15.18 MiB	Available
split_002.mp4	5/16/2025, 8:52:18 PM	Hot (Inferred)		Block blob	15.43 MiB	Available
split_003.mp4	5/16/2025, 8:52:21 PM	Hot (Inferred)		Block blob	15.27 MiB	Available
split_004.mp4	5/16/2025, 8:52:14 PM	Hot (Inferred)		Block blob	15.5 MiB	Available
split_005.mp4	5/16/2025, 8:52:16 PM	Hot (Inferred)		Block blob	14.2 MiB	Available
split_006.mp4	5/16/2025, 8:52:18 PM	Hot (Inferred)		Block blob	12.86 MiB	Available
split_007.mp4	5/16/2025, 8:52:13 PM	Hot (Inferred)		Block blob	14.32 MiB	Available
split_008.mp4	5/16/2025, 8:52:14 PM	Hot (Inferred)		Block blob	13.43 MiB	Available
split_009.mp4	5/16/2025, 8:52:17 PM	Hot (Inferred)		Block blob	14.89 MiB	Available
split_010.mp4	5/16/2025, 8:52:15 PM	Hot (Inferred)		Block blob	12.38 MiB	Available
split_011.mp4	5/16/2025, 8:52:19 PM	Hot (Inferred)		Block blob	13.12 MiB	Available
split_012.mp4	5/16/2025, 8:52:20 PM	Hot (Inferred)		Block blob	13.04 MiB	Available
split_013.mp4	5/16/2025, 8:52:15 PM	Hot (Inferred)		Block blob	13.61 MiB	Available
split_014.mp4	5/16/2025, 8:52:12 PM	Hot (Inferred)		Block blob	13.77 MiB	Available
split_015.mp4	5/16/2025, 8:52:21 PM	Hot (Inferred)		Block blob	12.67 MiB	Available
split_016.mp4	5/16/2025, 8:52:19 PM	Hot (Inferred)		Block blob	12.99 MiB	Available
split_017.mp4	5/16/2025, 8:52:19 PM	Hot (Inferred)		Block blob	183.46 KiB	Available

Η VideoProcessingFunction παρήγαγε τα εξής αποτελέσματα από την επεξεργασία των clips στο processed-stats container

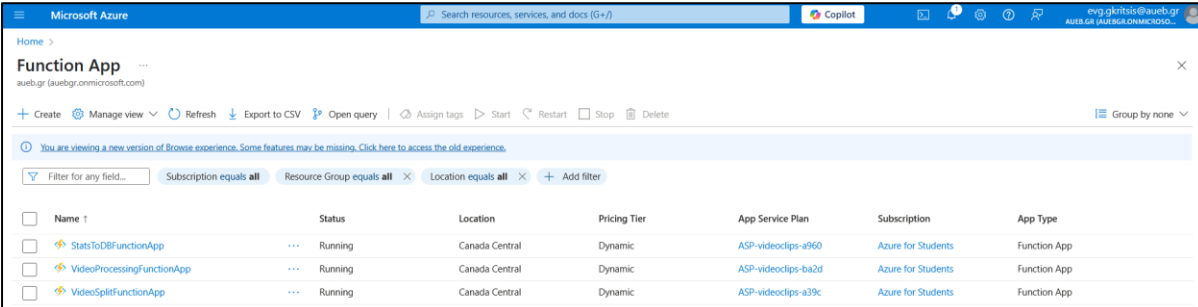


Name	Modified	Access tier	Archive status	Blob type	Size	Lease state
split_000_stats.json	5/21/2025, 1:13:42 PM	Hot (Inferred)		Block blob	83.93 KiB	Available
split_001_stats.json	5/21/2025, 1:16:26 PM	Hot (Inferred)		Block blob	71.78 KiB	Available
split_002_stats.json	5/21/2025, 1:19:11 PM	Hot (Inferred)		Block blob	81.25 KiB	Available
split_003_stats.json	5/21/2025, 1:21:58 PM	Hot (Inferred)		Block blob	81.4 KiB	Available
split_004_stats.json	5/21/2025, 1:24:56 PM	Hot (Inferred)		Block blob	82.61 KiB	Available
split_005_stats.json	5/21/2025, 1:27:35 PM	Hot (Inferred)		Block blob	76.82 KiB	Available
split_006_stats.json	5/21/2025, 1:30:04 PM	Hot (Inferred)		Block blob	58.71 KiB	Available
split_007_stats.json	5/21/2025, 1:32:32 PM	Hot (Inferred)		Block blob	80.02 KiB	Available
split_008_stats.json	5/21/2025, 1:35:02 PM	Hot (Inferred)		Block blob	78.44 KiB	Available
split_009_stats.json	5/21/2025, 1:37:45 PM	Hot (Inferred)		Block blob	86.01 KiB	Available
split_010_stats.json	5/21/2025, 1:40:11 PM	Hot (Inferred)		Block blob	66.5 KiB	Available
split_011_stats.json	5/21/2025, 1:45:05 PM	Hot (Inferred)		Block blob	68.77 KiB	Available
split_012_stats.json	5/21/2025, 1:48:05 PM	Hot (Inferred)		Block blob	74.76 KiB	Available
split_013_stats.json	5/21/2025, 1:50:49 PM	Hot (Inferred)		Block blob	84.8 KiB	Available
split_014_stats.json	5/21/2025, 1:53:33 PM	Hot (Inferred)		Block blob	83.45 KiB	Available
split_015_stats.json	5/21/2025, 1:56:22 PM	Hot (Inferred)		Block blob	78.65 KiB	Available
split_016_stats.json	5/21/2025, 1:59:08 PM	Hot (Inferred)		Block blob	88.1 KiB	Available

Για παράδειγμα βρέθηκαν τα εξής αποτελέσματα για το πρώτο split.

```
split_000_stats.json X
C: > Users > tzeni > Downloads > {} split_000_stats.json > [ ] vehicles > {} 0
1 {
2   "video_name": "split_000.mp4",
3   "processing_time": "2025-05-21T10:13:40.856047",
4   "total_vehicles": 365,
5   "vehicles": [
6     {
7       "id": 2,
8       "type": "car",
9       "direction": "outbound",
10      "speed": 81.20636221978874,
11      "timestamp": 0.28,
12      "position": [
13        785,
14        460
15      ],
16      "confidence": 0.7700573801994324
17    },
18    {
19      "id": 1,
20      "type": "car",
21      "direction": "outbound",
22      "speed": 95.25156115459458,
23      "timestamp": 0.24,
24      "position": [
25        424,
26        432
27      ],
28      "confidence": 0.8716236352920532
29    },
30    {
31      "id": 3,
32      "type": "car",
33      "direction": "outbound",
34      "speed": 134.7070179286856,
35      "timestamp": 0.48,
36      "position": [
37        284,
```

Ενδεικτικά παρουσιάζουμε και τις Azure Functions όπως φαίνονται στην πλατφόρμα Azure.



Function App						
aueb-gr (auebgr.onmicrosoft.com)						
+ Create Manage view Refresh Export to CSV Open query Assign tags Start Restart Stop Delete Group by none						
You are viewing a new version of Browse experience. Some features may be missing. Click here to access the old experience.						
Filter for any field... Subscription equals all Resource Group equals all Location equals all Add filter						
<input type="checkbox"/> Name 1	Status	Location	Pricing Tier	App Service Plan	Subscription	App Type
<input type="checkbox"/> StatsToDBFunctionApp	Running	Canada Central	Dynamic	ASP-videoclips-v960	Azure for Students	Function App
<input type="checkbox"/> VideoProcessingFunctionApp	Running	Canada Central	Dynamic	ASP-videoclips-ba2d	Azure for Students	Function App
<input type="checkbox"/> VideoSplitFunctionApp	Running	Canada Central	Dynamic	ASP-videoclips-a39c	Azure for Students	Function App

Microsoft Azure

Search resources, services, and docs (G+/)

Copilot

evg.gkritsis@auetb.gr
AUETB.GR (AUETB.GR/ONMICROSOFT)

Home > Function App > VideoSplitFunctionApp >

HttpVideoSplitFunction | Code + Test

VideoSplitFunctionApp

Code + Test Integration Function Keys Invocations Logs Metrics

Save Discard Refresh Test/Run Get function URL Disable Delete Upload Resource JSON Send us your feedback

This function has been edited through an external editor. Portal editing is disabled.

VideoSplitFunctionApp / HttpVideoSplitFunction / __init__.py

```
1 import logging
2 import azure.functions as func
3 import subprocess
4 import os
5 from azure.storage.blob import BlobServiceClient
6
7 def get_ffmpeg_path():
8     """Locate Ffmpeg binary with fallback paths"""
9     try:
10         subprocess.run(["ffmpeg", "-version"], check=True,
11                        stdout=subprocess.PIPE, stderr=subprocess.PIPE)
12         return "ffmpeg"
13     except:
14         bin_path = os.path.join(os.getcwd(), "bin", "ffmpeg")
15         if os.path.exists(bin_path):
16             return bin_path
```

App Insights Logs Log Level Stop Copy Clear Maximize Send us your feedback

Connected! You are now viewing logs of Function runs in the current Code + Test panel. To see all the logs for this function, please go to 'Logs' from the Function menu.

Microsoft Azure

Search resources, services, and docs (G+/)

Copilot

evg.gkritsis@auetb.gr
AUETB.GR (AUETB.GR/ONMICROSOFT)

Home > VideoSplitFunctionApp >

http_trigger_video_split | Code + Test

VideoSplitFunctionApp

Code + Test Integration Function Keys Invocations Logs Metrics

Save Discard Refresh Test/Run Get function URL Disable Delete Upload Resource JSON Send us your feedback

VideoSplitFunctionApp / function_app.py

```
14 ... stdout=subprocess.PIPE, stderr=subprocess.PIPE)
15     return "ffmpeg"
16 except:
17     # Check for Ffmpeg in the local 'bin' directory
18     local_ffmpeg = os.path.join(os.getcwd(), "bin", "ffmpeg")
19     if os.path.exists(local_ffmpeg):
20         return local_ffmpeg
21     raise Exception("Ffmpeg not found. Please ensure Ffmpeg is installed or included in your deployment")
22
23 @app.route(route="http_trigger_video_split")
24 def http_trigger_video_split(req: func.HttpRequest) -> func.HttpResponse:
25     logging.info('Video split function triggered.')
```

App Insights Logs Log Level Stop Copy Clear Maximize Send us your feedback

Test/Run

Input Output

HTTP response code 200 OK

HTTP response content

Video processed successfully

Run Close

Microsoft Azure

Search resources, services, and docs (G+/)

Copilot

evg.gkritsis@auetb.gr
AUETB.GR (AUETB.GR/ONMICROSOFT)

Home > Function App > VideoProcessingFunctionApp >

HTTPVideoProcessingFunction | Code + Test

VideoProcessingFunctionApp

Code + Test Integration Function Keys Invocations Logs Metrics

Save Discard Refresh Test/Run Get function URL Disable Delete Upload Resource JSON Send us your feedback

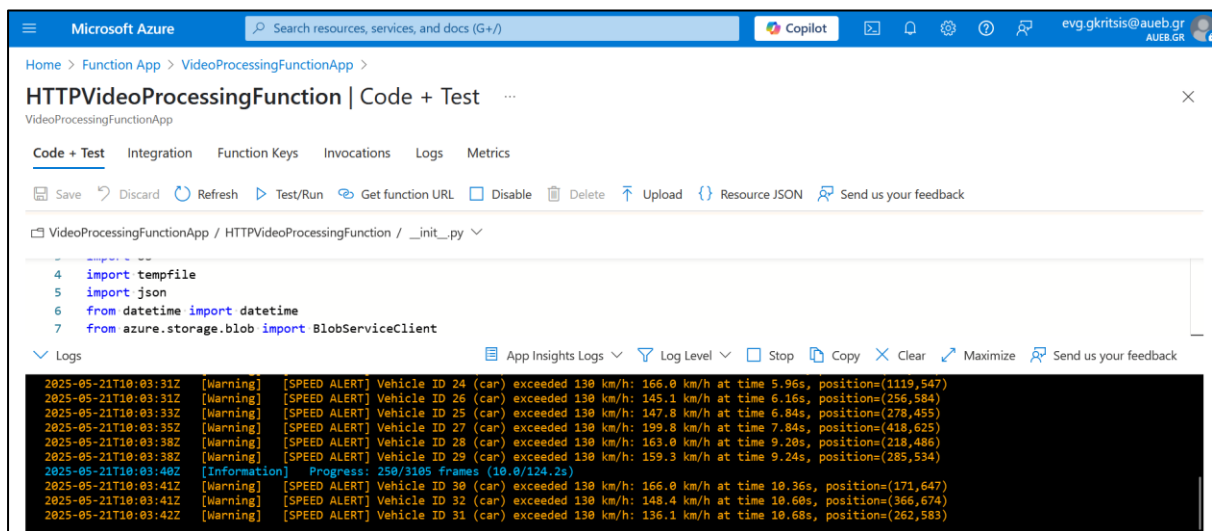
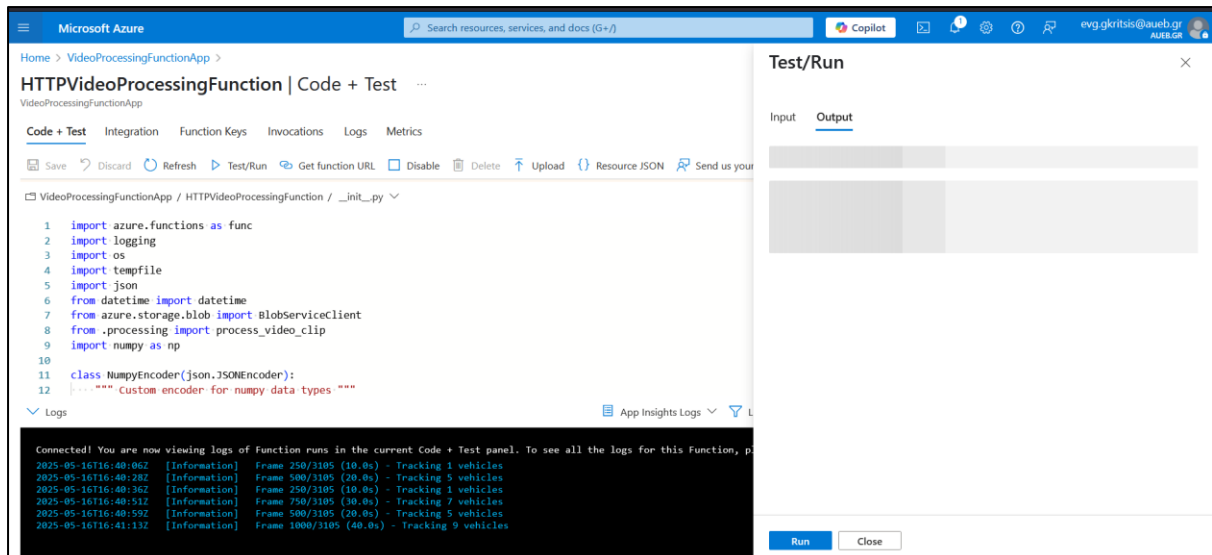
This function has been edited through an external editor. Portal editing is disabled.

VideoProcessingFunctionApp / HTTPVideoProcessingFunction / __init__.py

```
1 import azure.functions as func
2 import logging
3 import os
4 import tempfile
5 import json
6 from datetime import datetime
7 from azure.storage.blob import BlobServiceClient
8 from .processing import process_video_clip
9 import numpy as np
10
11 class NumpyEncoder(json.JSONEncoder):
12     """ Custom encoder for numpy data types """
13     def default(self, obj):
14         if isinstance(obj, (np.int_, np.intc, np.intp, np.int8,
15                             np.int16, np.int32, np.int64, np.uint8,
16                             np.uint16, np.uint32, np.uint64)):
```

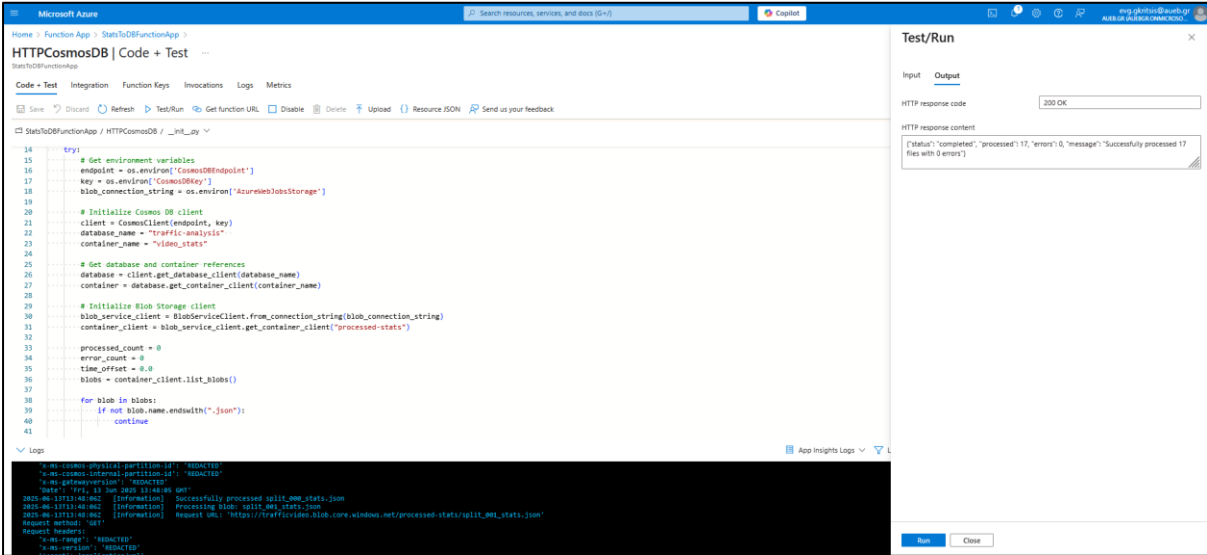
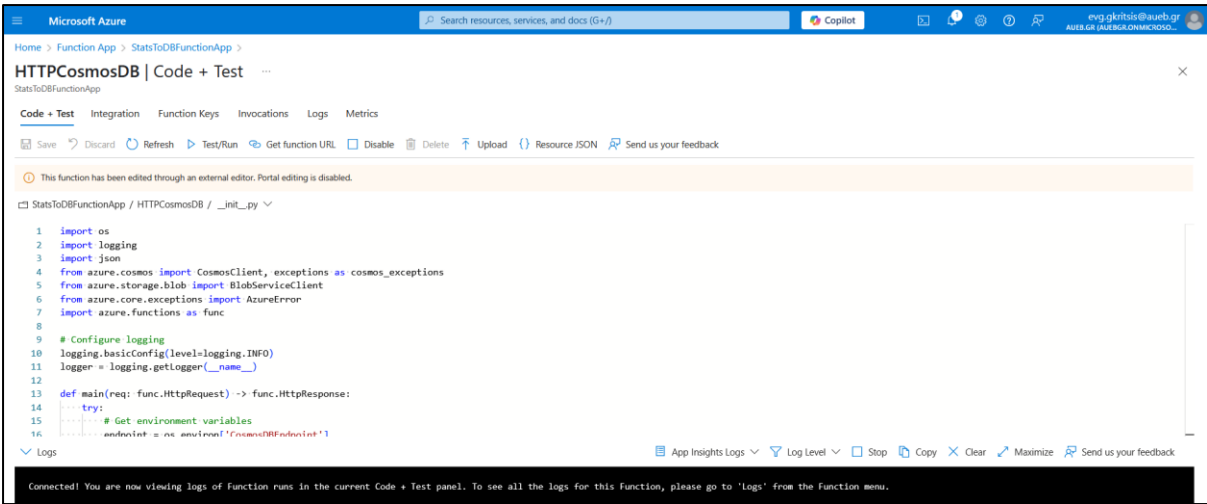
App Insights Logs Log Level Stop Copy Clear Maximize Send us your feedback

Connected! You are now viewing logs of Function runs in the current Code + Test panel. To see all the logs for this function, please go to 'Logs' from the Function menu.



Δημιουργήθηκε split 17 χωρίς οχήματα, καθώς αποτελεί το τέλος του βίντεο και δεν εμφανίζεται κάποιο όχημα.

```
[2025-05-21T10:59:07.495Z] Starting video processing: C:\Users\TZENIO-1\AppData\Local\Temp\tmpfmz877k1.mp4
[2025-05-21T10:59:07.690Z] Video properties - 1280x720 @ 25.0fps, Duration: 7.6s, Frames: 190
[2025-05-21T10:59:19.278Z] Processing completed. 0 unique vehicles in 11.6s
[2025-05-21T10:59:19.280Z] No vehicles detected in split 017.mp4
[2025-05-21T10:59:19.382Z] Executed 'Functions.HTTPVideoProcessingFunction' (Succeeded, Id=f928b3af-cbc5-4c70-9bb6-55bdf1538b38, Duration=1021752ms)
```



Test/Run

Input

Output

HTTP response code

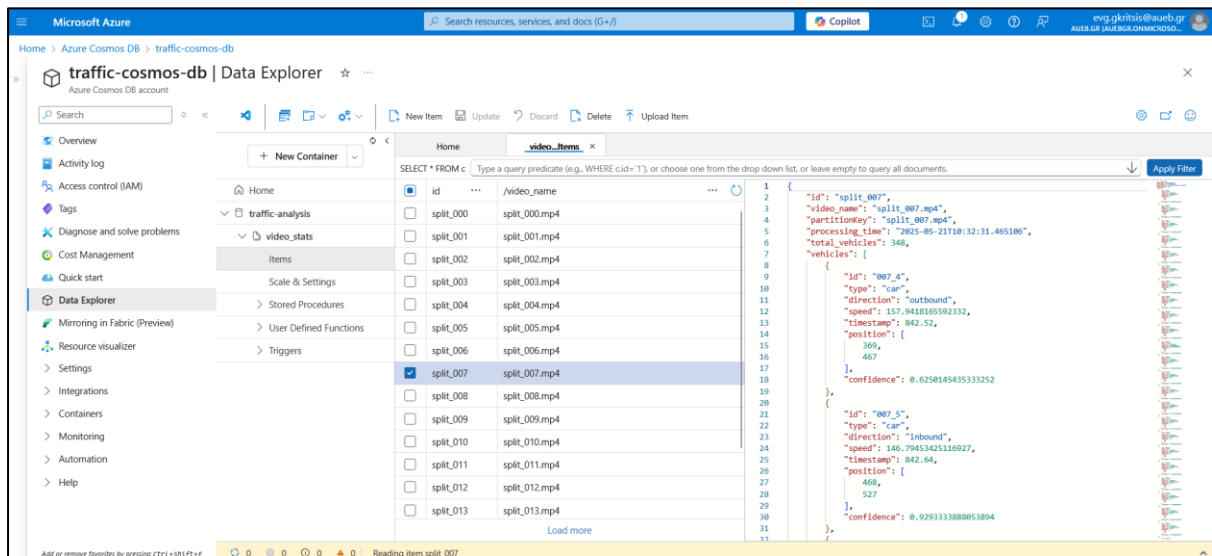
200 OK

HTTP response content

```
{
  "status": "completed",
  "processed": 17,
  "error": 0,
  "message": "Successfully processed 17 files with 0 errors"
}
```

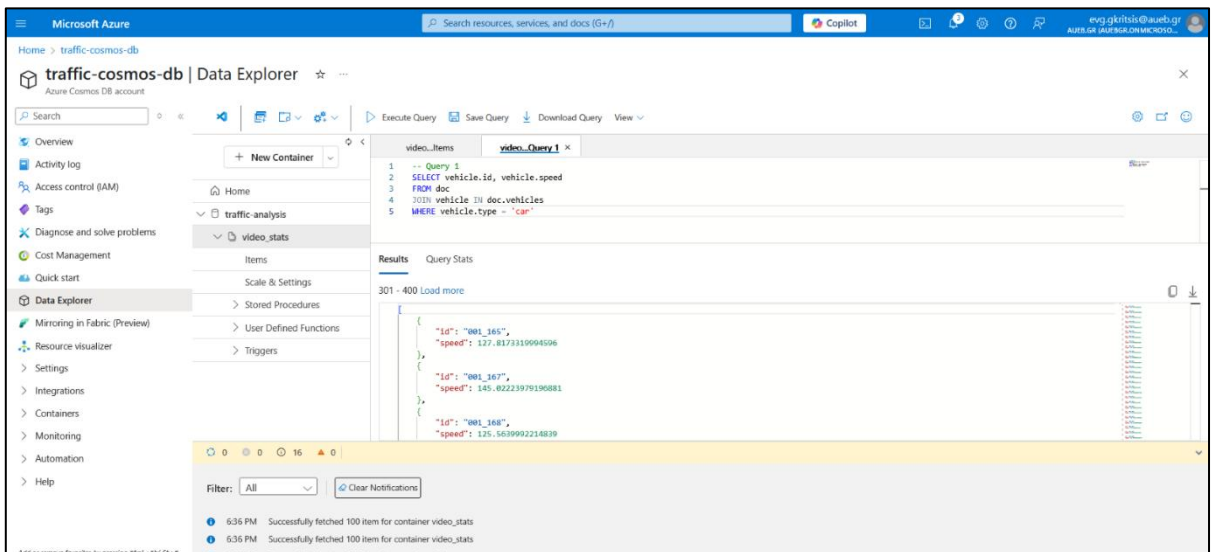
AutoClose

Βλέπουμε τα περιεχόμενα στην CosmosDB:



Ακολουθούν τέλος, τα στιγμιότυπα από την εκτέλεση των queries στην CosmosDB.

Query 1: Ταχύτητα με την οποία κινείται κάθε όχημα



Query 2: Αριθμός οχημάτων ανά ρεύμα

The screenshot shows the Microsoft Azure Data Explorer interface for the 'traffic-cosmos-db' account. The left sidebar contains navigation options like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Cost Management, Quick start, Data Explorer (selected), Mirroring in Fabric (Preview), Resource visualizer, Settings, Integrations, Containers, Monitoring, Automation, and Help. The main pane displays 'video_stats' with a 'video_items' container. A query is executed, showing the following SQL:

```
1 -- Query 2
2 SELECT vehicle.direction, COUNT(1) AS vehicle_count
3 FROM doc
4 JOIN vehicle IN doc.vehicles
5 GROUP BY vehicle.direction
```

The results are displayed in a JSON format:

```
{
  "direction": "outbound",
  "vehicle_count": 3562
},
{
  "direction": "inbound",
  "vehicle_count": 2391
}
```

The bottom status bar indicates 'Successfully fetched 2 item for container video_stats'.

Query 3: Αριθμός οχημάτων τα οποία έχουν υπερβεί το όριο ταχύτητας (αυτοκίνητα > 90χμ/ω και φορτηγά > 80χμ/ω).

The screenshot shows the Microsoft Azure Data Explorer interface for the 'traffic-cosmos-db' account. The left sidebar is the same as in the previous screenshot. The main pane displays 'video_stats' with a 'video_items' container. A query is executed, showing the following SQL:

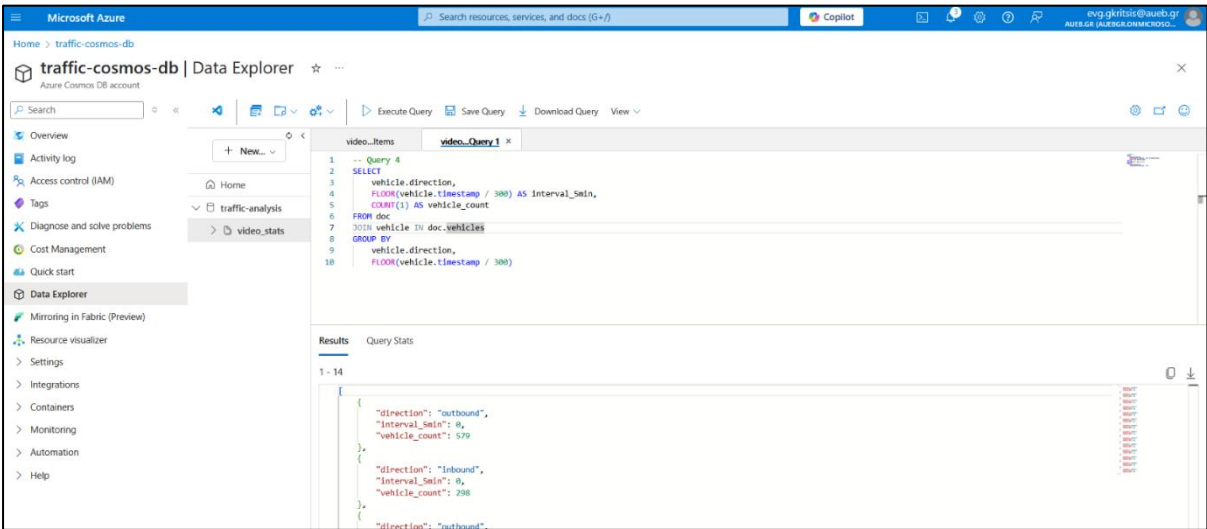
```
1 -- Query 3
2 SELECT VALUE COUNT(1)
3 FROM doc
4 JOIN vehicle IN doc.vehicles
5 WHERE (vehicle.type = 'Car' AND vehicle.speed > 90)
6 OR (vehicle.type = 'truck' AND vehicle.speed > 80)
```

The results are displayed in a JSON format:

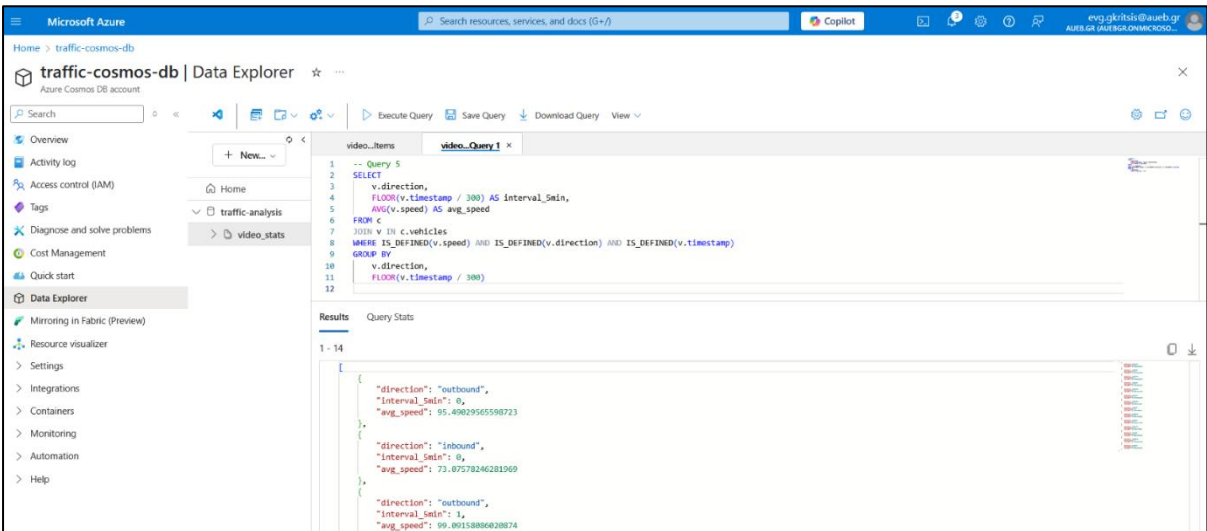
```
{
  "count": 3251
}
```

The bottom status bar indicates 'Successfully fetched 1 item for container video_stats'.

Query 4: Αριθμός οχημάτων ανά ρεύμα και ανά 5λεπτο.

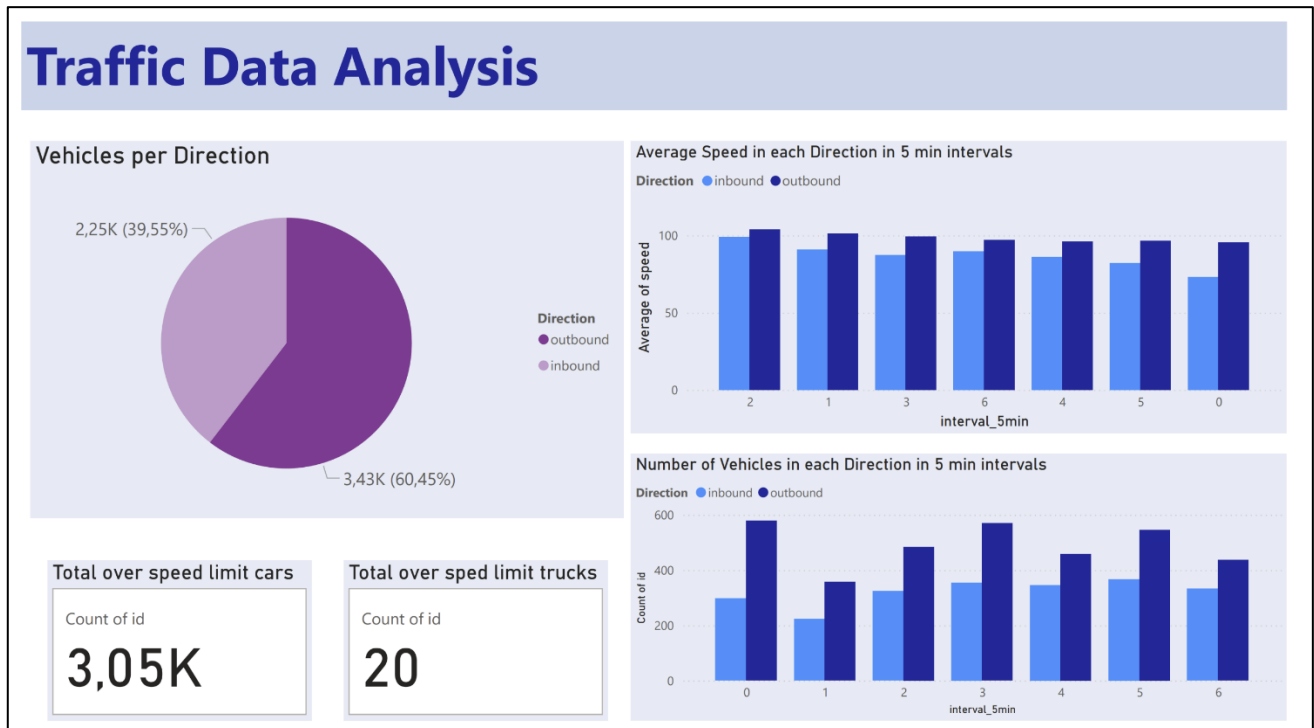


Query 5: Μέση ταχύτητα κίνησης ανά ρεύμα και ανά 5λεπτο



Αναφορά PowerBI

Προκειμένου να παρουσιαστούν τα τελικά αποτελέσματα της εργασίας, δημιουργήθηκε ένα καλαίσθητο PowerBI dashboard με τις σχετικές απαντήσεις των ερωτημάτων που τέθηκαν. Παρακάτω φαίνονται τα αποτελέσματα:



Εργαλεία

- **FFmpeg**. Το FFmpeg είναι ένα δωρεάν και ανοιχτού κώδικα λογισμικό, το οποίο αποτελείται από μια συλλογή βιβλιοθηκών και προγραμμάτων για τον χειρισμό αρχείων και ροών βίντεο, ήχου και άλλων πολυμέσων- <https://ffmpeg.org/>
- **Tiny YOLOv3**. Το μοντέλο Tiny YOLOv3 είναι ένα νευρωνικό δίκτυο για ανίχνευση αντικειμένων σε πραγματικό χρόνο. Είναι πολύ γρήγορο και ακριβές και πρόκειται για μια μικρότερη έκδοση του μοντέλου YOLOv3.- https://github.com/onnx/models/tree/main/validated/vision/object_detection_segmentation/tiny-yolov3
- **OpenCV**- Το OpenCV (Open-Source Computer Vision Library) είναι μια opensource βιβλιοθήκη συναρτήσεων για υπολογιστική όραση σε πραγματικό χρόνο- <https://opencv.org/>
- **Microsoft Azure** - Η Microsoft Azure είναι μια πλατφόρμα υπολογιστικού νέφους (cloud platform) που προσφέρει υπηρεσίες όπως αποθήκευση δεδομένων, επεξεργασία, τεχνητή νοημοσύνη και φιλοξενία εφαρμογών. - <https://azure.microsoft.com/en-gb>
- **Azure Functions** – Είναι η προσέγγιση της Microsoft στο serverless computing. - <https://azure.microsoft.com/en-us/products/functions>
- **Azure Blob Storage** - Είναι μια υπηρεσία αποθήκευσης αντικειμένων (object storage) της Microsoft, σχεδιασμένη για την αποθήκευση τεράστιων ποσοτήτων μη δομημένων δεδομένων. - <https://azure.microsoft.com/en-us/products/storage/blobs>
- **PowerBI** - Το Power BI είναι μια σύγχρονη πλατφόρμα Business Intelligence που αναπτύχθηκε από τη Microsoft, με σκοπό την οπτικοποίηση και ανάλυση δεδομένων. - <https://app.powerbi.com/home?experience=power-bi>
- **Cosmos DB** - Η Azure Cosmos DB είναι μια κατακευματισμένη βάση δεδομένων της Microsoft, σχεδιασμένη για εφαρμογές που απαιτούν υψηλή διαθεσιμότητα, χαμηλή καθυστέρηση και αυτόματη κλιμάκωση σε παγκόσμιο επίπεδο. - <https://azure.microsoft.com/en-us/products/cosmos-db>