ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

Μεταπτυχιακό Πρόγραμμα Σπουδών “Γεωπληροφορική”

“Αξιοποίηση εθελοντικής, γεωγραφικής πληροφορίας (VGI), σε περιβάλλον διαδικτυακής εφαρμογής, με αντικείμενο τον τουρισμό.”

Διπλωματική εργασία Ανδρέα Παπαγεωργίου

Αθήνα,
Ιανουάριος 2021

Τριμελής, εξεταστική επιτροπή:
Β. Βεσκούκης / Καθηγητής ΕΜΠ / Επιβλέπων

Ευχαριστίες

Εκπόνησα την διπλωματική μου εργασία με τον Καθηγητή Βασίλειο Βεσκούκη και τον υποψήφιο διδάκτορα Γιώργο Παπακυριακόπουλο. Ο κ. Βεσκούκης, μόλις του περιέγραψα τί με ενδιέφερε ως θέμα διπλωματικής, αμέσως με συνέστησε στον Γιώργο. Έπραξε πάρα πολύ σωστά!

Οι τρεις μας συνεργαστήκαμε εξαιρετικά, με έναν απλό και αποδοτικό τρόπο. Μία φορά την εβδομάδα, συζητούσαμε με τον Γιώργο την πρόοδό μου, και καθορίζαμε τα “παραδοτέα” της επόμενης εβδομάδας, ενημερώνοντας συχνά, τον κ. Βεσκούκη για το περιεχόμενο της συζήτησης. Όταν ήταν απαραίτητο, ο κ. Βεσκούκης συμμετείχε στις συζητήσεις, παρέχοντάς μου επιπλέον βοήθεια.

Καταλαβαίνει λοιπόν κανείς, ότι εκπονώντας την διπλωματική μου εργασία, είχα την τύχη, να συνεργαστώ ουσιαστικά με δύο ανθρώπους με σπάνια εμπειρία στην Γεωπληροφορική. Έμαθα και υλοποίησα την ιδέα, ότι όταν καλούμαι να λύσω ένα πρόβλημα, δεν είναι ανάγκη να το αντιμετωπίσω ευθέως. Πολλές φορές, είναι χρήσιμο να δημιουργήσω και λύσω απλούστερα προβλήματα, που μοιάζουν στο αρχικό! Ακόμη, έμαθα την αξία που έχει μια προσεγμένη παρουσίαση, και πώς να την δημιουργώ!

Δεν ξεχνώ τους Καθηγητές Γεώργιο Φώτη και Βύρωνα Νάκο, που με ώθησαν να επιλέξω το μεταπτυχιακό της Γεωπληροφορικής του ΕΜΠ, και με βοήθησαν να γίνω δεκτός σε αυτό!

Αφιερώνεται στον πατέρα και την μητέρα μου, που με τον τρόπο ζωής τους, καταδεικνύουν την αξία της υγείας, της τύχης, και της σταθερής αγάπης και αφοσίωσης στον σύντροφο.

Περίληψη

Η παρούσα εργασία, θίγει το πρόβλημα επιλογής προορισμού, στο πεδίο του τουρισμού. Προτείνουμε έναν ιστότοπο, βασισμένο στο WordPress και συνδεδεμένο με τα Google Maps APIs. Έχοντας ως κυρίαρχη την ιδέα των VGI, η εφαρμογή αντλεί τα δεδομένα της από τον χρήστη-εθελοντή. Η εφαρμογή αξιοποιεί την χωρική φύση του προβλήματος, κυρίως με την ενσωμάτωση της έννοιας της εγγύτητας. Κύρια χωρική οντότητα είναι η τεθλασμένη γραμμή, η οποία συμβολίζει μία διαδρομή στον χώρο. Επομένως, η εφαρμογή δίνει έμφαση στο τί υπάρχει κοντά στην εκάστοτε διαδρομή, αλλά και σε χαρακτηριστικά αυτής καθεαυτής της διαδρομής, εν προκειμένω, της θερμοκρασίας και της κλίσης της. Τα παραπάνω καθιστούν χρήσιμη την εφαρμογή σε εύρος δραστηριοτήτων όπως το απλό βάδισμα που συνδυάζεται με επίσκεψη σημείων ενδιαφέροντος (pois), ή και η άθληση.

Περιεχόμενα

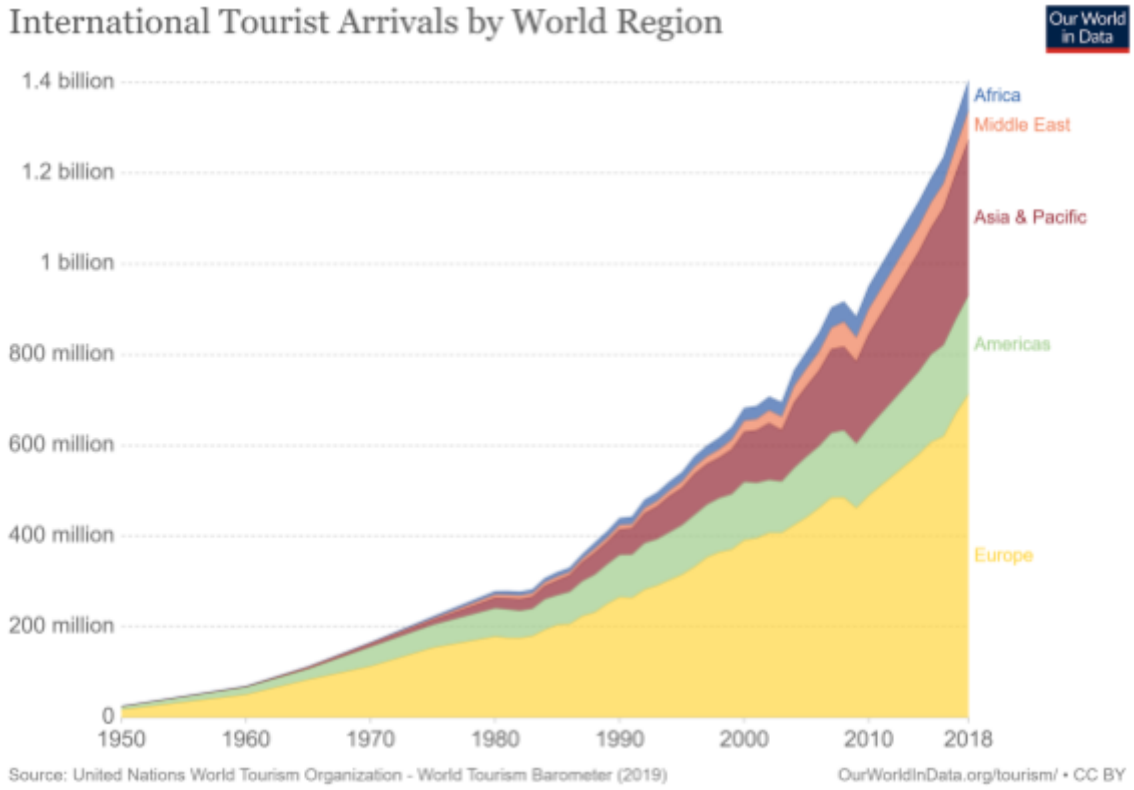
Πίνακας πινάκων

Πίνακας εικόνων

Εισαγωγή

Το πρόβλημα της εργασίας

Σε όλο τον κόσμο, ο τουρισμός παράγει κοινωνικο-οικονομικά οφέλη αντίστοιχα τομέων όπως οι εξαγωγές πετρελαίου, διατροφικών προϊόντων, και αυτοκινήτων. Επιπλέον, το οικονομικό μέγεθος του τουρισμού, παρουσιάζει πολύ μεγάλο ρυθμό αύξησης [1]. Η Εικόνα 1 φανερώνει την προαναφερθείσα αύξηση, και την κατανομή των αφίξεων ανά ήπειρο:



Εικόνα 1: Κατανομή αφίξεων τουριστών, ανά ήπειρο, στον χρόνο.
(Πηγή: United Nations World Tourism Organization)

Η περιγραφείσα κατάσταση, σαφώς εμπεριέχει το πρόβλημα της επιλογής προορισμού. Το πρόβλημα αυτό, μπορεί κάλλιστα να υποστηρίξει η τεχνολογία, χάρη στις σημαντικές δυνατότητες και λειτουργικότητες που παρέχει. Επίσης, η κουλτούρα χρήσης της τεχνολογίας στον τουρισμό, υπάρχει, εν μέρει διότι την καλλιεργούν δημοφιλείς εφαρμογές που αναφέρουμε στις Σχετικές εφαρμογές. Έτσι, διαδικτυακά μέσα με ιδιαιτερότητα, δηλαδή, που η ιδέα τους και η χρήση τους ξεφεύγουν από το σύνηθες, χωρίς αυτό να σημαίνει μεγάλη καινοτομία, μπορούν να τραβήξουν την προσοχή ικανού αριθμού ανθρώπων, και έτσι, να υποστηρίξουν το μέγεθος και την ανάπτυξη του τουρισμού.

[1] “Why Tourism? | UNWTO.”

Σχετικές εφαρμογές

Ενδεικτικές εφαρμογές που αφορούν τον τουρισμό, και όχι μόνον (Google Maps), είναι οι:

- Google Maps
- Tripadvisor
- Trivago

Το Google Maps παρέχει δυνατότητες:

- Οπτικοποίησης ξενοδοχείων σε χάρτη
- Σύγκρισης κόστους διαμονής, μέσω οπτικοποίησης του κόστους διανυκτέρευσης σε χάρτη
- Διασταύρωσης πληροφοριών με άλλες εφαρμογές, χάρη σε αντίστοιχες διασυνδέσεις που παρέχει το Google Maps

Τα Tripadvisor και Trivago ανήκουν σε μία πολυπληθή κατηγορία εφαρμογών που σχεδόν αγνοούν την χωρική διάσταση του τουρισμού, και παρέχουν δυνατότητες:

- Εξέτασης κόστους διανυκτέρευσης
- Αναζήτησης ξενοδοχείου με κριτήρια που στην πλειονότητά τους δεν είναι χωρικά

Παρατηρούμε ότι οι εφαρμογές αυτές αξιοποιούν σε μικρό βαθμό το χωρικό της φύσης του προβλήματος της επιλογής προορισμού. Επίσης, τους λείπει σημαντική πληροφορία που μπορεί να παρέξει ο χρήστης που γνωρίζει τον εκάστοτε τόπο. Στην συνέχεια, καταγράφουμε την πρότασή μας.

Η πρόταση

Η τρέχουσα εργασία προτείνει μία διαδικτυακή εφαρμογή η οποία έχει σκοπό να:

- Βοηθήσει στην επιλογή μέρους προς επίσκεψη, σε οποιοδήποτε μέρος του κόσμου, αρκεί να
  υπάρχει αρκετή/κατάλληλη για αυτό, πληροφορία
- παρέξει ψυχαγωγία με την μορφή “ψηφιακού τουρισμού”

Η εφαρμογή αφορά εν δυνάμει τουρίστες, και τουρίστες που ήδη βρίσκονται στον προορισμό τους. Επίσης, αφορά “ψηφιακούς επισκέπτες”. Στην κατηγορία αυτή ανήκει και η κατηγορία εκείνων που αδυνατούν να επισκεφθούν κάποιο μέρος, για λόγους οικονομικούς, υγείας, ή άλλους.
Η ιδιαιτερότητα της εφαρμογής προκύπτει από την χρήση του ίχνους, ως κεντρικής, χωρικής οντότητας. Παράγωγη της ανωτέρω ιδιαιτερότητας είναι η έννοια της κίνησης, την οποία υπονοεί το γραμμικό του ίχνους. Η εφαρμογή μπορεί να καλύψει *βάδισμα* και *jogging*.
Ο χρήστης θα μπορεί να:

- Σχεδιάσει και ανεβάσει ένα ίχνος με multimedia content.
- Διανύσει ένα ίχνος.
- Απαντήσει ερωτήματα

- Αναζητήσει ίχνη τα οποία ικανοποιούν ένα σετ κριτηρίων.
- Προβάλει την μηκοτομή ενός ίχνους
- Προβάλει τα pois που ανήκουν σε ακτίνες που ο ίδιος επιλέγει, πέριξ του ίχνους επιλογής.

Πιθανώς, η κυριότερη ιδέα της εφαρμογής είναι η Εθελοντική Γεωγραφική Πληροφορία (ΕΓΠ, VGI), δηλαδή η παροχή δυνατότητας ανεβάσμος γεωγραφικής πληροφορίας από τους χρήστες της εφαρμογής, και η αξιοποίησή της.

Γνώσεις και ορισμοί

Ορισμοί

Label	**Ορισμός**
API	*“API είναι μια οντότητα λογισμικού, η οποία μειώνει την πολυπλοκότητα που απαιτεί η επίτευξη ορισμένων υπολογισμών. Ο χρήστης, χρησιμοποιώντας σχετικά απλές εντολές, ενεργοποιεί στοιχεία του API, τα οποία αντιστοιχούν σε πολύπλοκους υπολογισμούς.”* [2]
Cloud computing	Το cloud computing, αναφέρεται στην προσβασιμότητα του χρήστη, σε κάποια υπολογιστική υποδομή, μέσω internet. [3]
Crowdsourcing και VGI (Volunteered, Geographic Information)	Αναζητώντας βιβλιογραφία περί crowdsourcing, σχηματίζουμε την εντύπωση ότι το μεγαλύτερο μέρος της αφορά διαδικασίες ανάπτυξης λογισμικού. Όμως, στο πλαίσιο της εργασίας, ενδιαφέρει η υποκατηγορια crowdsourcing, το VGI. Η ουσία της είναι η λήψη γεωγραφικών δεδομένων από το "κοινό". [4] Σαφώς, περιλαμβάνει και άλλα ζητήματα όπως η διασφάλιση της ποιότητας των γεωγραφικών δεδομένων.
Content-management system (CMS)	*“Content-management system, είναι λογισμικό δημιουργίας και διαχείρισης ψηφιακού περιεχομένου.”*[5]
Front end	Είναι η επιφάνεια επαφής της εφαρμογής με τον χρήστη.

[2] “Introduction to Web APIs.”
[3] “What Is Cloud Computing?”
[4] Goodchild, “CITIZENS AS SENSORS: THE WORLD OF VOLUNTEERED GEOGRAPHY”; Batty et al., “Map Mashups, Web 2.0 and the GIS Revolution.”
[5] “Content Management System.”

Distributed, web GIS [6]	*"Distributed, web GIS, είναι το GIS που:* ● *Αντλεί δεδομένα από πολλές πηγές* ● *Αξιοποιεί διάφορα services* ● *Μπορεί να αξιοποιεί ποικιλία user interfaces, όπως desktop, κινητά, wearables."*
Google Maps Platform	*"Το Google Maps Platform είναι ένα σύνολο API, τα οποία επιτρέπουν την τοποθέτηση χαρτών σε ιστοσελίδες ή εφαρμογών σε κινητό τηλέφωνο, ή την λήψη δεδομένων που περιλαμβάνουν τα Google Maps."* [7]
JSON και GeoJSON	*"JSON είναι ένα ελαφρύ format, κατάλληλο για ανταλλαγή δεδομένων. Περιλαμβάνει ένα σύνολο ζευγών name-value."* [8] *"GeoJSON είναι επίσης format, που βασίζεται στο format "JSON", και αναπαριστά χωρικές οντότητες (σημεία, γραμμές, πολύγωνα, και άλλα), και μη-χωρικά χαρακτηριστικά τους."* [9]
Mashup	Mashup είναι μια κατηγορία εφαρμογών. Προκειμένου να πούμε ότι μια εφαρμογή είναι mashup, πρέπει, ή να χρησιμοποιεί δεδομένα από διάφορες πηγές, ή/και να συνδυάζει διάφορα λογισμικά.
Multimedia content	Multimedia content Multimedia content, μπορεί να είναι ένα βίντεο, εικόνες, υπερκείμενο, και λοιπά.
Open data	*"Προκειμένου να χαρακτηρίσουμε ένα σετ δεδομένων ως σετ ανοιχτών δεδομένων, πρέπει να έχει, μεταξύ άλλων, τις ακόλουθες ιδιότητες [10]:* ● *Availability and Access: Το εκάστοτε σετ δεδομένων πρέπει να διατίθεται ως ενότητα, δηλαδή, όχι σε κομμάτια. Επίσης, πρέπει να έχει βολική, και τροποποιήσιμη μορφή.*

[6] Vescoukis, "5. Αρχιτεκτονικές Εφαρμογών Διαδικτύου."
[7] "Google Maps Platform FAQ."
[8] "JSON."
[9] "GeoJSON—ArcGIS Online Help | Documentation."
[10] "What Is Open Data?"

	• *Re-use and Redistribution: Τα δεδομένα πρέπει να συνοδεύονται όπου όρους χρήσης, που επιτρέπουν την αναδιανομή τους, καθώς και την ανάμειξή τους με άλλα δεδομένα.* • *Universal Participation: Τα δεδομένα, πρέπει να μπορεί να τα χρησιμοποιήσει οιοσδήποτε."*
Open Geospatial Consortium (OGC)	*"Πρόκειται για οργανισμό με μέριμνα για την [11]:* • *Διευκόλυνση της εύρεσης χωρικής πληροφορίας* • *Αύξηση της προσβασιμότητας στην χωρική πληροφορία* • *Διαλειτουργικότητα της χωρικής πληροφορίας"*
Openweathermap.org	Στο openweathermap.org, υπάρχουν APIs που παρέχουν πληροφορία για τον καιρό. Εκείνο που ενδιαφέρει την παρούσα εργασία, είναι το One-call API. Το ακόλουθο url αποτελεί γενίκευση του εκάστοτε αιτήματος: https://api.openweathermap.org/data/2.5/onecall?lat={lat}&lon={lon}&exclude={part}&appid={API key}, Όπου: Lat, και lon: το γεωγραφικό πλάτος και μήκος, το σημείο για το οποίο ζητάμε δεδομένα Part: παράμετρος που παίρνει τιμές current, minutely, hourly, daily, και alert, διαχωρισμένες με κόμμα. Οι τιμές που επιλέγουμε, εξαιρούν από την απόκριση, αντίστοιχα δεδομένα.
Ανάλυση χώρου	Η ανάλυση χώρου προσπαθεί να εξηγήσει φαινόμενα του χώρου, ή να εκτιμήσει την εξέλιξη αυτών.
Ανάλυση εγγύτητας	Η ανάλυση εγγύτητας, εντοπίζει τα αντικείμενα συγκεκριμένης κατηγορίας, τα οποία ανήκουν σε ζώνης επιρροής, γύρω από κάποια αντικείμενο. Η απόσταση μπορεί να

[11] "The Home of Location Technology Innovation and Collaboration | OGC."

	είναι ευκλείδεια, δικτυακή, Manhattan, ή άλλου τύπου.
Γεωγραφικό, Πληροφοριακό Σύστημα (ΓΠΣ) (Geographic, Information System) (GIS)	Ένα ΓΠΣ παρέχει δυνατότητες [12]: ● Εντοπισμού προβλημάτων ● Παρακολούθησης μεταβολών ● Διαχείρισης και αντίδρασης σε συμβάντα όπως οι φυσικές καταστροφές ● Εκτίμησης (forecast) ● Εντοπισμού τάσεων
Διαλειτουργικότητα (interoperability)	Υπάρχουν αρκετοί ορισμοί της διαλειτουργικότητας. Μπορούμε ίσως να πούμε ότι η διαλειτουργικότητα χαρακτηρίζει δύο ή περισσότερες οντότητες, όταν αυτές μπορούν να ανταλλάξουν δεδομένα, και να τα επεξεργαστούν σύμφωνα με συμφωνημένες διαδικασίες.[13]
Ελεύθερο λογισμικό (free, or open-source software)	Ελεύθερο λογισμικό (free, or open-source software) “Ελεύθερο, λέμε το λογισμικό το οποίο επιτρέπει στον χρήστη[14]: ● Να το χρησιμοποιήσει όπως αυτός επιθυμεί ● Να το αντιγράψει ● Να το διανείμει ● Να το τροποποιήσει”
Μεταδεδομένα	Μεταδεδομένα είναι δεδομένα που αφορούν κάποια δεδομένα. Για παράδειγμα, αν έχουμε ένα αρχείο με θερμοκρασίες ανά διοικητική διαίρεση, η ημερομηνία τελευταίας τροποποίησης του αρχείου είναι μεταδεδομένο.
Μηκοτομή	Μηκοτομή είναι μία τεθλασμένη γραμμή, επί κατακόρυφου επιπέδου, η οποία παρουσιάζει την υψομετρία μιας γραμμής του χώρου. Δηλαδή, η μηκοτομή έχει έναν άξονα των χ, κι

[12] “What Is GIS? | Geographic Information System Mapping Technology.”
[13] Brownsword et al., “Current Perspectives on Interoperability.”
[14] “Free and Open Source Software.”

	έναν άξονα των y. Ο άξονας των χ, συνήθως περιλαμβάνει την απόσταση του σημείου από την αρχή της γραμμής, επί οριζοντίου επιπέδου, και ο άξονας των y περιλαμβάνει τα υψόμετρα.

Πίνακας 1: Ορισμοί

Γνώσεις και πληροφορίες

Google-Maps, Javascript API: δείγμα “elevation-along-path”
Το δείγμα “elevation along path”, του Google-Maps, Javascript API (Παράρτημα Α : δείγμα “elevation along path”.), δείχνει πώς μπορούμε να ζητήσουμε ένα ίχνος, προβεβλημένο σε χάρτη, καθώς και την μηκοτομή του ίχνους.

Google-Maps, Javascript API: υπηρεσία (service) “Elevation”.
Η υπηρεσία “Elevation”, του Google-Maps, Javascript API, παρέχει δυνατότητα λήψης υψομέτρου συγκεκριμένου σημείου, μεταξύ άλλων, παρόμοιων δυνατοτήτων. Μπορούμε να λάβουμε το υψόμετρο συγκεκριμένου σημείου, με url της μορφής:
`http://maps.googleapis.com/maps/api/elevation/outputFormat?parameters` ,
Όπου:

- outputFormat: το format της απόκρισης (JSON ή XML)
- Parameters: λέξη την οποία αντικαθιστούμε με έκφραση της μορφής “locations: x, y”, όπου x και y, οι συντεταγμένες του σημείου.

Google-Maps, Places API
Το Places API της Google, παρέχει, μεταξύ άλλων, δυνατότητα λήψης των θέσεων αντικειμένων συγκεκριμένης κατηγορίας (type), γύρω από καθορισμένο σημείο και απόσταση. Μαζί με τις θέσεις, το API αποστέλλει και άλλες, χρήσιμες πληροφορίες. Πραγματοποιούμε το αίτημα με url της μορφής:
`https://maps.googleapis.com/maps/api/place/nearbysearch/output?parameters`,
Όπου:

- Output: το format της απόκρισης
- Parameters: Παράμετρος του url, την οποία αντικαθιστούμε με ένα σύνολο παραμέτρων, υποχρεωτικών και μη. Υποχρεωτικές, είναι οι παράμετροι:
    - Key: Κλειδί API
    - Location: Οι συντεταγμένες της θέσης γύρω από την οποία αναζητούμε πληροφορία
    - Radius: Η ακτίνα αναζήτησης

Google-Maps, Visualization API
Το Visualization API, της πλατφόρμας Google Maps, παρέχει, μεταξύ άλλων, δυνατότητα χρήσης διαγραμμάτων όπως η μηκοτομή.

Εθελοντική Γεωγραφική Πληροφορία (ΕΓΠ, VGI)
ΕΓΠ, είναι γεωγραφική πληροφορία που προσφέρει ο χρήστης, δωρεάν. Ένα από τα κρίσιμα θέματα σχετικά με την ΕΓΠ, είναι τα *όρια* της ποιότητάς της. Υπάρχουν παραδείγματα που δείχνουν ότι τα όρια αυτά είναι άκρως ικανοποιητικά, και μπορούν να προσεγγίσουν την ποιότητα της πληροφορίας που παρέχουν επαγγελματίες. Επομένως, με κατάλληλες κινήσεις, η ΕΓΠ μπορεί και να αντικαταστήσει την πληροφορία επαγγελματιών. [15]
Ουσιαστικά, δεν αδιαφορούμε για τα κακώς κείμενα που κατά καιρούς προκύπτουν, ως ΕΓΠ, όμως αυτό δεν διαψεύδει το ότι μπορεί να αποτελέσει επάξια λύση, και μάλιστα οικονομική, έστω υπό όρους.
Πέραν της οικονομικότητάς της, η ΕΓΠ ξεπερνά τις δυνατότητες της πληροφορίας που μπορούν να παρέξουν επαγγελματίες, όσον αφορά την γεωγραφική κάλυψη, αφού μπορεί να καλύψει και περιοχές στις οποίες το καθεστώς απαγορεύει την παραγωγή πληροφορίας από θεσμικά όργανα.

Τεχνολογίες

Label	**Περιεχόμενο**
CSS	*"Η CSS είναι μηχανισμός που επιτρέπει την μορφοποίηση των εγγράφων που συνιστούν τον ιστό."* [16]
HTML	*"Η HTML είναι η τεχνολογία που καθορίζει την δομή και το περιεχόμενο του ιστού. Επιτρέπει την σύνδεση μεταξύ εγγράφων του ιστού."* [17]
JavaScript	*"JavaScript είναι γλώσσα προγραμματισμού, με την οποία μπορούμε να φτιάξουμε δυναμικό web περιεχόμενο."* [18] Συνυπάρχει και δεν υποκαθιστά τις τεχνολογίες HTML και CSS.
PHP	*"Η PHP είναι γλώσσα προγραμματισμού, ελεύθερης διάθεσης, η οποία λύνει*

[15] Goodchild, "CITIZENS AS SENSORS: THE WORLD OF VOLUNTEERED GEOGRAPHY."
[16] "Cascading Style Sheets."
[17] "HTML."
[18] "What Is JavaScript?"

	προβλήματα web development, και μπορεί να μπει μέσα στην HTML.” [19]

Πίνακας 2: Τεχνολογίες

Έννοιες και ορισμοί WordPress

Label	**Περιεχόμενο**
WordPress	Το WordPress είναι CMS, του οποίου τον ορισμό έχουμε παρουσιάσει παραπάνω. Παρακάτω, καταγράφουμε ορισμούς που αφορούν σημαντικές, για την εργασία μας, οντότητες του WordPress.
Custom fields: Keys και values	Ένα key και το αντίστοιχο value, συνθέτουν ένα custom field. Το custom field είναι μεταδεδομένα που αντιστοιχούν σε ένα post.
Page και Post	Υπάρχουν δύο μορφές/τρόποι εισαγωγής περιεχομένου στο WordPress. Τα pages και τα posts. Ίσως η κύρια διαφορά μεταξύ των δύο, είναι τα posts έχουν χρονολογική αναφορά. Μία ακόμη σημαντικά διαφορά, είναι ότι την οργάνωση των posts την επιτυγχάνουν τα taxonomies, ενώ των pages την επιτυγχάνουν σχέσεις παιδιού-γονέα. [20]
Plugin	Plugin είναι κώδικας PHP, ο οποίος επεκτείνει την standard λειτουργικότητα του WordPress. Το WordPress Plugin Repository (αποθετήριο) παρέχει μία σειρά από plugins.
Plugin “CSS-JS-PHP”	Το plugin “CSS-JS-PHP”, δίνει την δυνατότητα εκτέλεσης προγραμμάτων, με το αποτέλεσμά τους να εμφανίζεται μέσα σε posts. Το πρόγραμμα μπορεί να περιλαμβάνει CSS, Javascript, HTML, και PHP.

[19] “PHP: Preface - Manual.”
[20] “Pages.”

Plugin “Progress Map”	Το plugin “Progress Map”, δίνει, μεταξύ άλλων, δυνατότητες: ● Εύκολης και γρήγορης δημιουργίας αντικειμένων χαρτών, και προβολής των αντίστοιχων χαρτών σε posts. ● Δημιουργίας διαδρομών, με markers, και επισύναψη multimedia content στα markers. ● Δημιουργίας λίστας κριτηρίων αναζήτησης posts, τα οποία, αν αντιστοιχούν σε γεωμετρικές οντότητες όπως σημεία και γραμμές, τότε, η αναζήτηση δίνει αντίστοιχα αποτελέσματα.
Shortcode	Shortcode είναι κειμενο λίγων χαρακτήρων, το οποίο αντικαθιστά κώδικα, πιθανώς πολλών γραμμών. Σημαντική λειτουργικότητα που παρέχει, είναι ότι εκτελεί ένα κομμάτι κώδικα, όταν ανανεώνουμε ένα post. Πέραν αυτής, με τα shortcodes αποφεύγουμε το υπερσυσσώρευση πληροφορίας στο παράθυρο διαμόρφωσης του περιεχομένου ενός post.
Taxonomy	“Taxonomy” είναι κατηγορία που περιλαμβάνει τις κατηγορίες “tag” και “category”. Αφορούν posts και όχι pages. Διευκολύνουν τον εντοπισμό συναφούς με συγκεκριμένο post, περιεχομένου, το οποίο ανήκει στον ιστότοπό μας. [21]

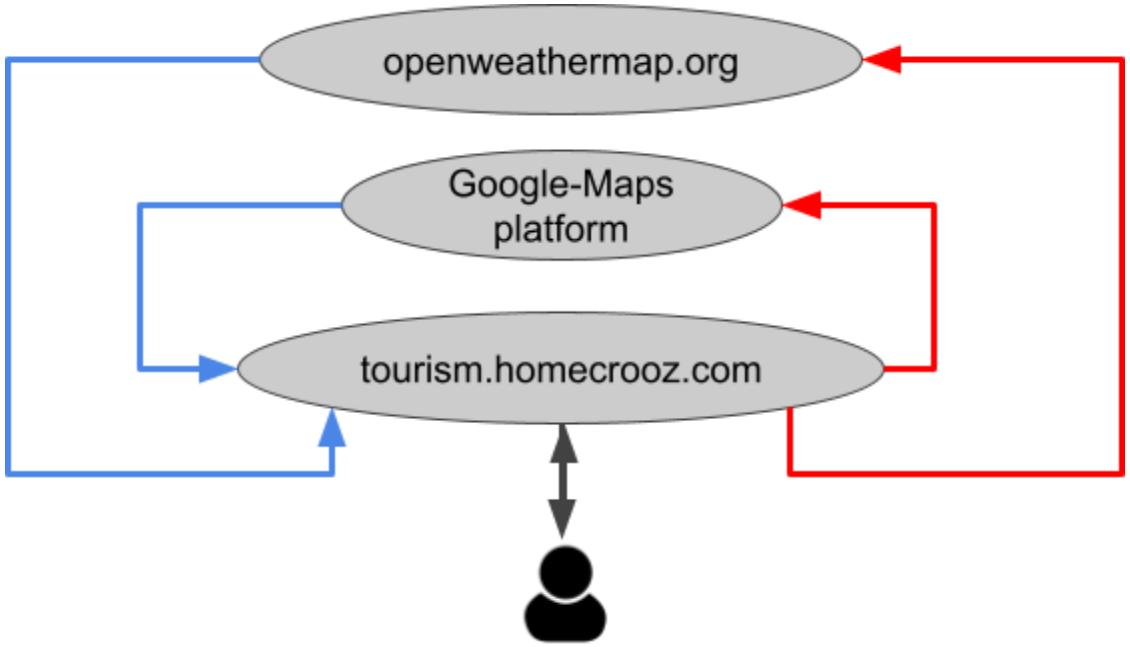
Πίνακας 3: Έννοιες και ορισμοί WordPress

[21] “Taxonomies.”

Αρχιτεκτονική της εφαρμογής

Η αρχιτεκτονική της εφαρμογής

Στην εφαρμογή μας, το WordPress επικοινωνεί με την πλατφόρμα “Google Maps” και με το openweathermap.org, σύμφωνα με την Εικόνα 2:

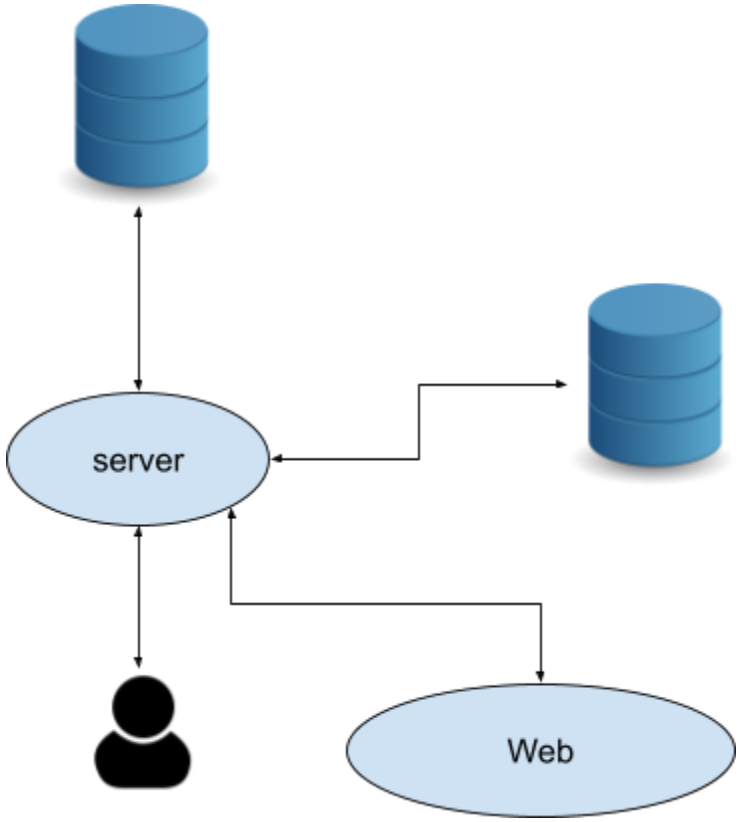


Εικόνα 2: Γενική αρχιτεκτονική της εφαρμογής.

Ο client μπορεί να αντιστοιχεί σε οποιαδήποτε συσκευή, φορητή ή μη, που διαθέτει δυνατότητα χρήσης browser.

Επιμέρους αρχιτεκτονική: Uploading και εμφάνιση ίχνους

Βασική ιδιότητα της εφαρμογής, είναι η δυνατότητα ανεβάσματος ίχνους από τον χρήστη, μαζί με υλικό multimedia. Τις λεπτομέρειες ανεβάσματος ίχνους θα τις αναφέρουμε σε παρακάτω κεφάλαιο και εδάφιο. Εδώ, σημειώνουμε ότι ο χρήστης υποβάλει τις συντεταγμένες των σημείων του ίχνους, και αναρτά σε αυτά υλικό, είτε από το διαδίκτυο, είτε από τον σκληρό του δίσκο, είτε από την βάση δεδομένων του WordPress. Το ανέβασμα των συντεταγμένων, επιτρέπει το Progress Map plugin. Η Εικόνα 3 περιλαμβάνει τα ανωτέρω:



Εικόνα 3: Αρχιτεκτονική uploading και εμφάνισης ίχνους

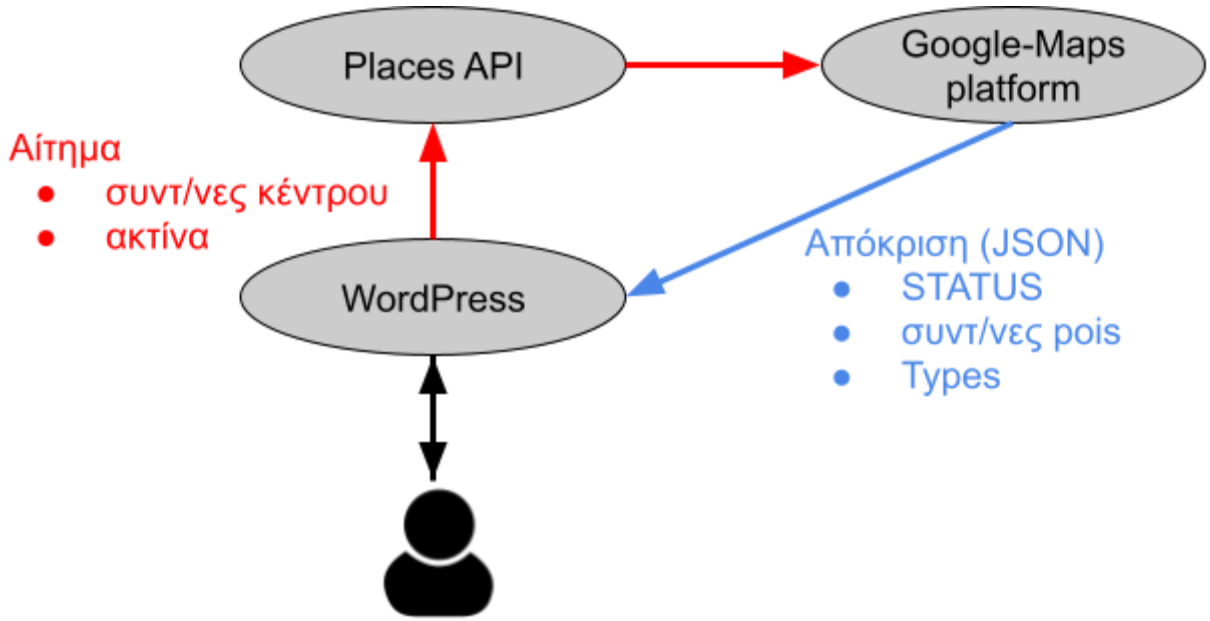
Επιμέρους αρχιτεκτονική: Ανανέωση τιμών

Places API

Η εφαρμογή χρησιμοποιεί αντικείμενα του Places API, ώστε να ελέγξει, εάν, γύρω από κάθε ίχνος της εφαρμογής, στην απόσταση που καθορίζει ο χρήστης, υπάρχει τουλάχιστον ένα αντικείμενο από τις κατηγορίες που έχουμε καθορίσει. Αυτό, το απαντά η τιμή της μεταβλητής “STATUS”, του JSON απόκρισης. Οι κατηγορίες περιλαμβάνουν επιλεγμένα, Google types, σύμφωνα με την εξής λίστα:

- Φαγητό: bakery, meal takeaway, restaurant
- Αστυνομία: police
- Καφές: bar, cafe
- Σημείο ενδιαφέροντος για τουρίστες: tourist attraction
- Θρησκευτικό κτίριο: church, hindu temple, mosque, synagogue

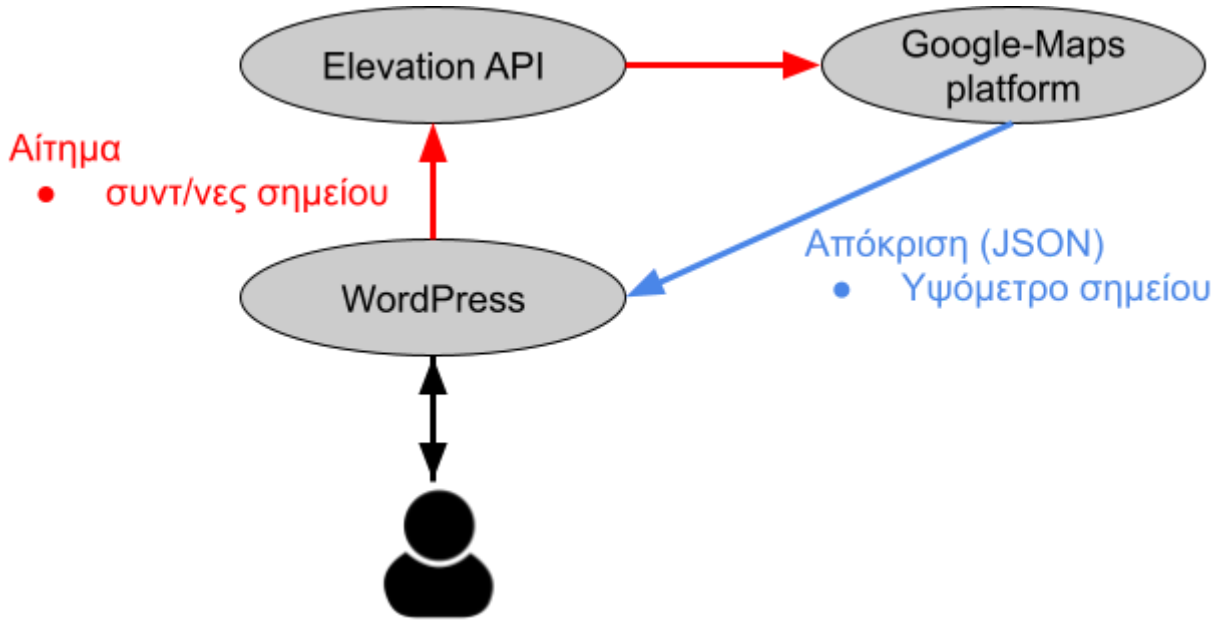
Η Εικόνα 4 περιγράφει την διαδικασία:



Εικόνα 4: Έλεγχος ύπαρξης pois, σε Google-Maps platform, μέσω Places API.

Elevation API

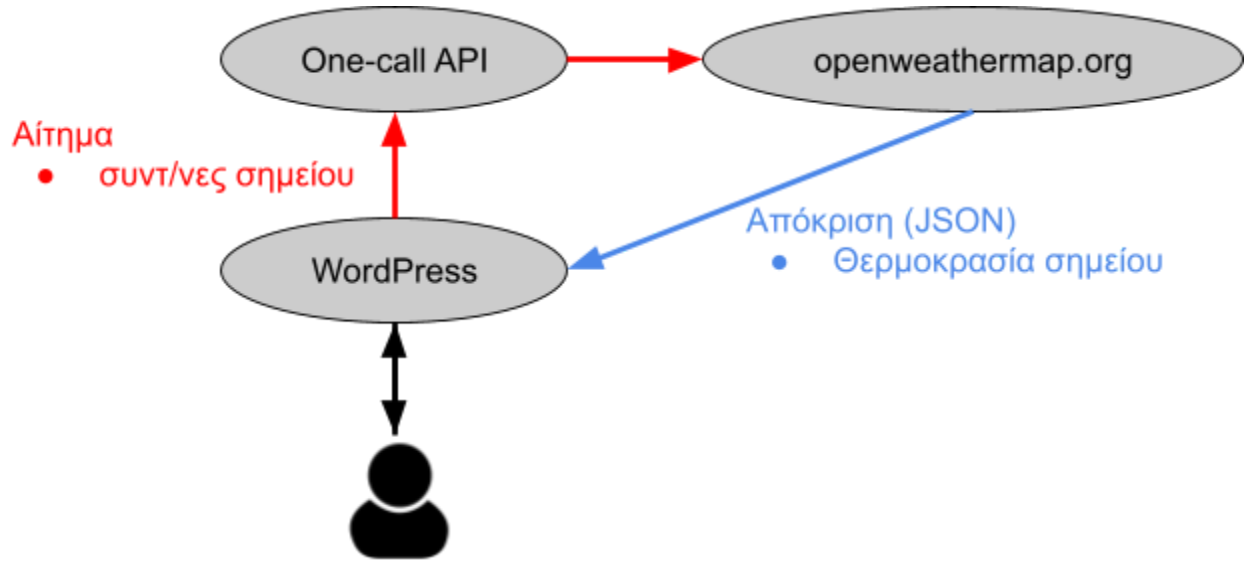
Η εφαρμογή χρησιμοποιεί αντικείμενα του Elevation API, ώστε να λάβει τα υψόμετρα των σημείων των ιχνών. Τα υψόμετρα αυτά, τα αξιοποιεί στον υπολογισμό της μέσης κλίσης των νέων ιχνών, εφόσον ο χρήστης κάνει την αντίστοιχη επιλογή. Η Εικόνα 5 περιγράφει την διαδικασία:



Εικόνα 5: Λήψη υψομέτρων για υπολογισμό κλίσης, από Google-Maps Platform, μέσω Elevation API.

One-call API

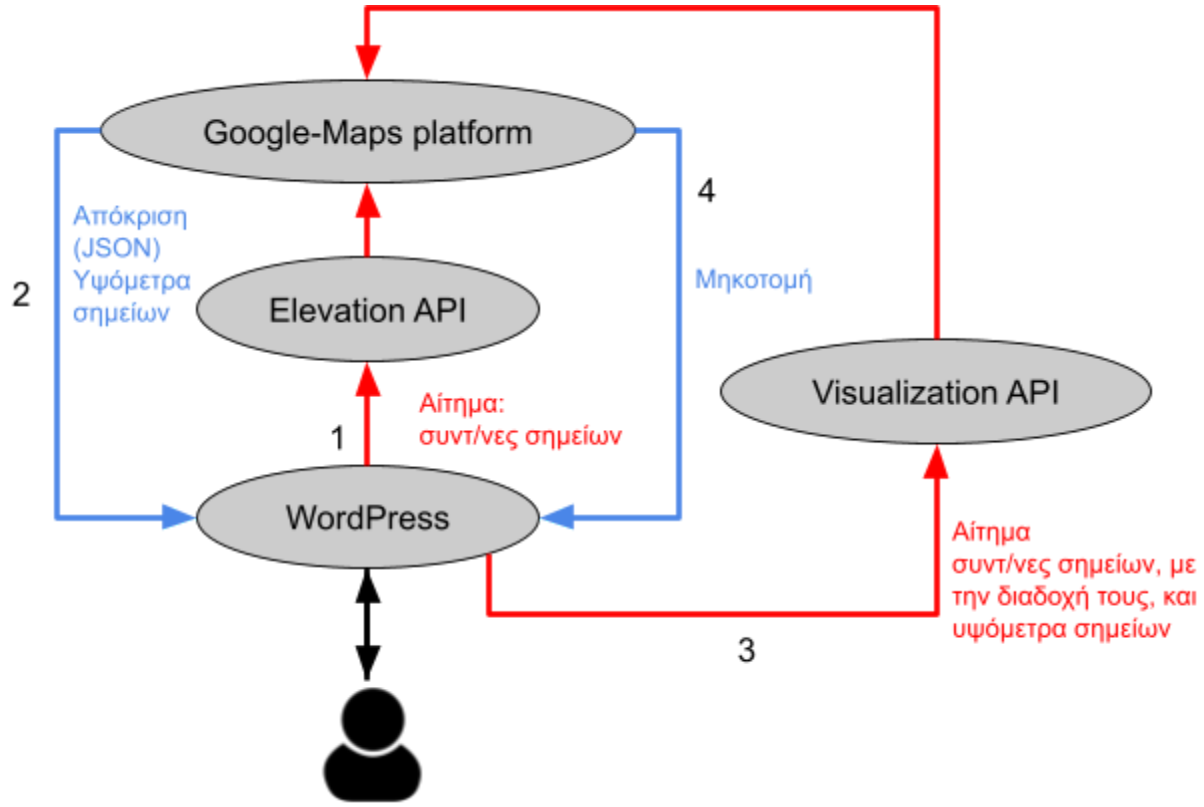
Η εφαρμογή χρησιμοποιεί αντικείμενα του One-Call API, ώστε να λάβει την θερμοκρασία κάθε ίχνους, την στιγμή ενεργοποίησης της αντίστοιχης επιλογής, ή παραπλήσια στιγμή, σύμφωνα με την Εικόνα 6:



Εικόνα 6: Λήψη θερμοκρασίας από openweathermap.

Επιμέρους αρχιτεκτονική: Λήψη μηκοτομής από Google-Maps platform.

Η εφαρμογή λαμβάνει υψόμετρα σημείων ενός ίχνους, μέσω του Elevation API, σύμφωνα με το Elevation API. Τα υψόμετρα αυτά, τα αποστέλει, μαζί με τις γεωγραφικές συντεταγμένες τους, και, μέσω του Visualization API, λαμβάνει και προβάλει την μηκοτομή του ίχνους, όπως δείχνει η Εικόνα 7:



Εικόνα 7: Λήψη μηκοτομής από Google-Maps platform.

Επιμέρους αρχιτεκτονική: Εμφάνιση pois

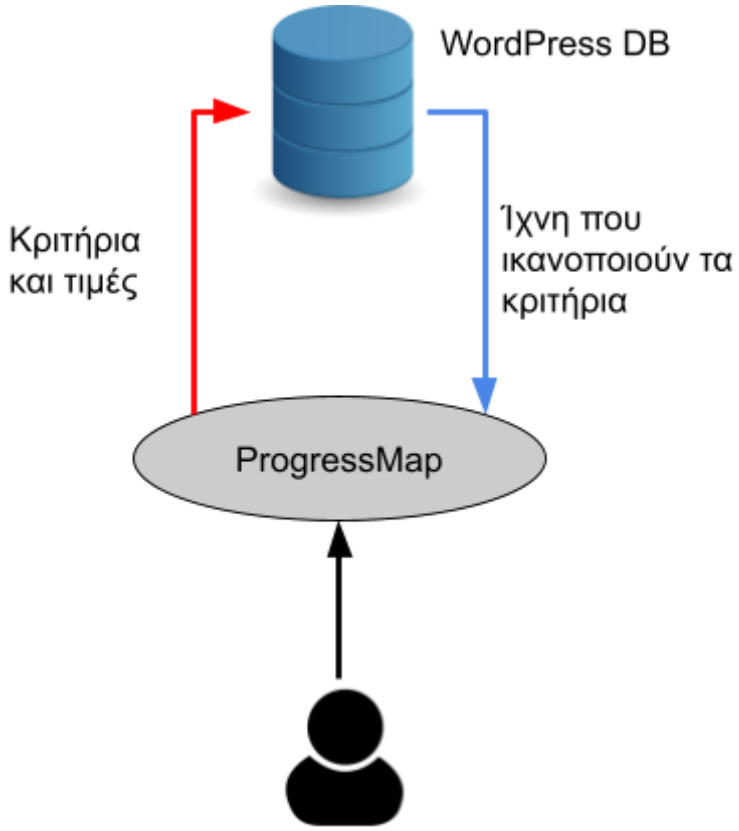
Την αρχιτεκτονική της δυνατότητας εμφάνισης pois, περιγράφει η Εικόνα, παραπάνω. Με την διαφορά, ότι παραπάνω ενδιαφέρει η μεταβλητή “STATUS”, ενώ εδώ ενδιαφέρουν όλες οι θέσεις, κάθε κατηγορίας ενδιαφέροντος, εντός της καθορισμένης ακτίνας, ώστε να μπορέσουμε να τις προβάλουμε σε χάρτη.

Επιμέρους αρχιτεκτονική: Εμφάνιση pois, και μηκοτομής ίχνους

Στην υλοποίηση της εφαρμογής, την δυνατότητα εμφάνισης pois εντός επιθυμητών ακτινών ανά κατηγορία pois, και την προβολή μηκοτομής του ίχνους, παρέχει ένα πρόγραμμα, και όχι δύο, όπως υπονοεί η παράθεση δύο διαφορετικών Εικόνων (για λόγους επεξηγηματικής αξίας), παραπάνω. Σημειώνουμε ότι οι δύο διαδικασίες είναι ανεξάρτητες, επομένως, δεν υπάρχει ζήτημα καθορισμού διαδοχής, η οποία να περιλαμβάνει βήματα και από τις δύο διαδικασίες.
Επιπλέον, καταγράφουμε ότι, όπως θα δείξουμε στο κεφάλαιο Μεθοδολογίας και υλοποίησης της εφαρμογής, η εφαρμογή λαμβάνει εικονογραφικά σύμβολα των pois, από το http://kml4earth.appspot.com .

Επιμέρους αρχιτεκτονική: Λήψη ιχνών που ικανοποιούν κάποια κριτήρια

Η αρχιτεκτονική υποβολής κριτηρίων και τιμών, και λήψης ιχνών που ικανοποιούν τα κριτήρια και τις τιμές, λαμβάνει χώρα κυρίως εντός WordPress, με εξωτερική παρέμβαση μόνον εκ μέρους του χρήστη. Όπως δείχνει η Εικόνα 8, ο χρήστης επιλέγει τα κριτήρια, τα υποβάλει μέσω Progress Map plugin στην βάση του WordPress, και λαμβάνει τα ίχνη, στην οθόνη του.



Εικόνα 8: Λήψη ιχνών που ικανοποιούν τα κριτήρια.

Μεθοδολογία της εφαρμογής

Χωρίζουμε το τρέχον κεφάλαιο, σε πέντε ενότητες, αντίστοιχες των λειτουργικοτήτων της εφαρμογής.

Δεδομένα, πηγές, γεωγραφική αναφορά.

Η εφαρμογή χρησιμοποιεί δύο κύριες πηγές δεδομένων, και ορισμένες δευτερεύουσες. Οι κύριες, είναι το Google-Maps Platform, και το openweathermap. Οι δευτερεύουσες είναι ο σκληρός δίσκος του εκάστοτε χρήστη και το web.
Το τί δεδομένα λαμβάνουμε από το κάθε Google API, και από το openweathermap, το αναφέρουμε στο εδάφιο "Γνώσεις και πληροφορίες". Από τον σκληρό δίσκο του εκάστοτε χρήστη, καθώς και από το web, μπορούμε να λάβουμε multimedia content, ως επένδυση κάποιου ίχνους, αν το επιθυμεί ο χρήστης. Σχεδίαση και ανέβασμα ίχνους, με multimedia content.
Η ακρίβεια της οριζοντιογραφίας του Google Maps, δίνει μέσο τετραγωνικό σφάλμα της τάξεως των δέκα μέτρων, [22] το οποίο αφήνει ανεπηρέαστη την ποιότητα της εφαρμογής. Το μέσο τετραγωνικό σφάλμα της υψομετρίας είναι ακόμη χαμηλότερο [23], συνεπώς, ικανοποιεί και αυτό την εφαρμογή.
Όσον αφορά την γεωγραφική αναφορά της εφαρμογής, η δεύτερη επιτρέπει το ανέβασμα ιχνών σε οποιαδήποτε περιοχή του κόσμου.

Επιλογή WordPress. Παρελκόμενες τεχνολογίες.

Οι λόγοι που επιλέξαμε το WordPress, είναι οι εξής:

- Αποτελεί CMS, και, επομένως παρέχει πολλές δυνατότητες σε διαχειριστή και χρήστη.
- Παρέχει ευχέρεια επέκτασης ή επέκτασης της εφαρμογής, μεταξύ άλλων, μέσω ενσωμάτωσης υπαρχουσών δυνατοτήτων από έτοιμα plugins.
- Καλύπτει πολλές συσκευές, μιας και το χρησιμοποιούμε μέσω browser, και διαθέτει προσαρμοστικότητας της επιφάνειας χρήστη, αναλόγως τον τύπο της συσκευής. (tablet, smartphone, etc.)

Η χρήση του WordPress, το οποίο προβάλει περιεχόμενο στον Web, οδηγεί σε χρήση των τεχνολογιών HTML, για μορφοποίηση, JavaScript, για υπολογισμούς στον client, και PHP, για υπολογισμούς στον server, μεταξύ των οποίων έμμεση ή άμεση συναλλαγή με την βάση δεδομένων που περιλαμβάνει η πλατφόρμα.

[22] Ubukawa, "An Evaluation of the Horizontal Positional Accuracy of Google and Bing Satellite Imagery and Three Roads Data Sets Based on High Resolution Satellite Imagery."
[23] El-Ashmawy, "Investigation of the Accuracy of Google Earth Elevation Data."

Σχεδίαση και ανέβασμα ίχνους με multimedia content.

Όσον αφορά την σχεδίαση και το ανέβασμα ίχνους με multimedia content, την επιτρέπει το Progress Map plugin. Συγκεκριμένα, το Progress Map plugin δίνει στον χρήστη την δυνατότητα να “καρφιτσώσει” ένα σημείο σε χάρτη του Google, αντιστοιχώντας το σε ένα post. Η δυνατότητα μετάδοσης, στην εφαρμογή, της πληροφορίας ότι το σημείο x ανήκει, με άλλα σημεία, στο ίχνος y, προκύπτει από τις βασικές λειτουργικότητες του WordPress, μέσω ανάρτησης tag του τύπου “ίχνος y”, στο σημείο x. Το Progress Map plugin αντιμετωπίζει τις συντεταγμένες ως μεταδεδομένα, και τις εκχωρεί σε δύο fields τα οποία, το plugin δημιουργεί και ονομάζει με ονομασία κοινή, για όλα τα posts. (codespacing_progress_map_lat και codespacing_progress_map_lng) Το multimedia content, ο χρήστης το βάζει μέσω βασικών λειτουργικοτήτων του WordPress, σε σημεία του ίχνους, στον χάρτη του ίχνους, ή στην επιφάνεια του post που αντιστοιχεί σε σημείο, ή σε ίχνος.

Εμφάνιση ίχνους σε χάρτη, και προβολή προσαρτημένου, multimedia content.

Σχετικά με την εμφάνιση ίχνους σε χάρτη, και την προβολή προσαρτημένου σε αυτό, multimedia content, και πάλι την παρέχει το Progress Map plugin. Με το plugin, μπορούμε να δημιουργήσουμε χάρτη που να δείχνει μόνο το ίχνος ενδιαφέροντος. Μπορούμε να ενώσουμε τα σημεία με την διαδοχή που επιθυμούμε. Επίσης, με κατάλληλη ρύθμιση του plugin, μπορούμε να προβάλουμε, κλικάροντας σημείο του ίχνους, multimedia όπως βίντεο και εικόνες, εντός του περιβάλλοντος του χάρτη.

Διάνυση ίχνους

Την διάνυση ίχνους, βλέποντας παράλληλα τον χάρτη και την τρέχουσα θέση, την διαθέτει και πάλι, κατάλληλη ρύθμιση του Progress Map plugin. Η βασική ρύθμιση είναι η ενεργοποίηση του εικονιδίου “Google-Street View”, επί του χάρτη.

Απάντηση ερωτημάτων εγγύτητας

Όπως αναφέραμε στο “Places API”, οι κατηγορίες pois που ενδιαφέρουν την εφαρμογή περιλαμβάνουν επιλεγμένα, Google types, σύμφωνα με την εξής λίστα:

- Φαγητό: bakery, meal takeaway, restaurant
- Αστυνομία: police
- Καφές: bar, cafe
- Σημείο ενδιαφέροντος για τουρίστες: tourist attraction
- Θρησκευτικό κτίριο: church, hindu temple, mosque, synagogue

Ο χρήστης μπορεί να ρωτήσει την εφαρμογή, εάν, για όλα τα ίχνη της εφαρμογής, και στην απόσταση που ο ίδιος καθορίζει, για όποια κατηγορία τον ενδιαφέρει, υπάρχει τουλάχιστον ένα αντικείμενο. Έτσι, για παράδειγμα, αν ο χρήστης δώσει απόσταση χιλίων μέτρων, για την

κατηγορία φαγητού, τότε, για κάθε ίχνος για το οποίο υπάρχει τουλάχιστον ένα αντικείμενο της κατηγορίας, εντός ακτίνας χιλίων μέτρων, το αντίστοιχο field, σε κάθε ίχνος, θα λάβει τιμή “at least one”, ενώ αν δεν υπάρχει, θα λάβει την τιμή “none”. Σημειώνουμε ότι η ακτίνα εξέτασης έχει κέντρο, το κέντρο βάρους του εκάστοτε ίχνους.
Από την παρούσα λειτουργικότητα απουσιάζει το Progress Map plugin, όμως συμμετέχει το CSS-JS-PHP plugin. Με ένα λοιπόν πρόγραμμα, που έχουμε τοποθετήσει με την μορφή shortcode, και ως διαχείριση του ιστότοπου σε ένα post, με τίτλο του post, “temperature”, μπορούμε, με προβολή του post, και βάζοντας σε φόρμα HTML, στις κατηγορίες που μας ενδιαφέρουν, τις ακτίνες που επιθυμούμε, και χωρίς εμπλοκή σε διαδικασίες προγραμματισμού, να λάβουμε απάντηση, και στην συνέχεια να την δούμε στο field που εκπροσωπεί την κατηγορία ενδιαφέροντος.

Ανανέωση τιμής θερμοκρασίας

Ο χρήστης μπορεί να λάβει την τιμή της τρέχουσας, ή σχεδόν τρέχουσας θερμοκρασίας, σε όλα τα ίχνη. Καταγράφουμε ότι την θερμοκρασία την λαμβάνουμε για σημείο που αντιπροσωπεύει το ίχνος, που ο χρήστης που έχει ανεβάσει το εκάστοτε ίχνος, έχει βάλει. Εκτός εξαιρετικών περιπτώσεων μεγάλης έκτασης ίχνους, ή ίχνους με μεγάλες υψομετρικές διαφορές, η θερμοκρασία αυτή αντιπροσωπεύει επαρκώς το ίχνος.
Σχετικά με τα εργαλεία που εξυπηρετούν την παρούσα λειτουργικότητα, χρησιμοποιούμε το CSS-JS-PHP plugin, με τον τρόπο που το κάνουμε στα ερωτήματα εγγύτητας. Η τρέχουσα περίπτωση είναι απλούστερη, διότι απαλλασσόμαστε από την ανάγκη εκχώρησης της τιμών σε φόρμα HTML. Το μόνο που κάνουμε είναι να προβάλουμε το post που περιλαμβάνει το shortcode θερμοκρασίας.

Υπολογισμός μέσης κλίσης

Ο χρήστης μπορεί να υπολογίσει την τιμής μέσης κλίσης, ιχνών που ανέβασε, ή ιχνών που ανέβασαν άλλοι χρήστες, χωρίς να υπολογίσουν την μέση κλίση τους.
Υπολογίζουμε την μέση κλίση ως τον μέσο όρο των κλίσεων κάθε στοιχειώδους ευθύγραμμου τμήματος του ίχνους, εκτός αν το ίχνος αποτελεί μόνο ένα ευθύγραμμο τμήμα. Υπολογίζουμε την κλίση κάθε ευθύγραμμου τμήματος ως το πηλίκον της υψομετρικής διαφοράς, δια το μήκος του τμήματος.

Αναζήτηση ίχνους, με κριτήρια εγγύτητας, θερμοκρασίας, και μέσης κλίσης

Οι λειτουργικότητες απάντησης ερωτημάτων εγγύτητας, ανανέωσης τιμών θερμοκρασίας, και υπολογισμού μέσης κλίσης, τις οποίες περιγράψαμε παραπάνω, αποτελούν αυτόνομες λειτουργικότητες. Όμως, εξυπηρετούν και την λειτουργικότητα αναζήτησης ιχνών υπό αντίστοιχα κριτήρια. Δηλαδή, προκειμένου να αναζητήσουμε ίχνη στα οποία, εντός ακτίνας χιλίων μέτρων υπάρχει τουλάχιστον ένα αντικείμενο της κατηγορίας φαγητού, και τα οποία έχουν κλίση μεταξύ ενός και πέντε τοις εκατό, θα πρέπει, για όλα μας τα ίχνη, να έχουμε τιμή στο πεδίο που δηλώνει

αν υπάρχει τουλάχιστον ένα αντικείμενο της κατηγορίας φαγητού, εντός ακτίνας χιλίων μέτρων, και να έχουμε τιμή στο πεδίο κλίσης.
Την δυνατότητα τέτοιων αναζητήσεων, την παρέχει το Progress Map plugin.

Προβολή μηκοτομής ίχνους

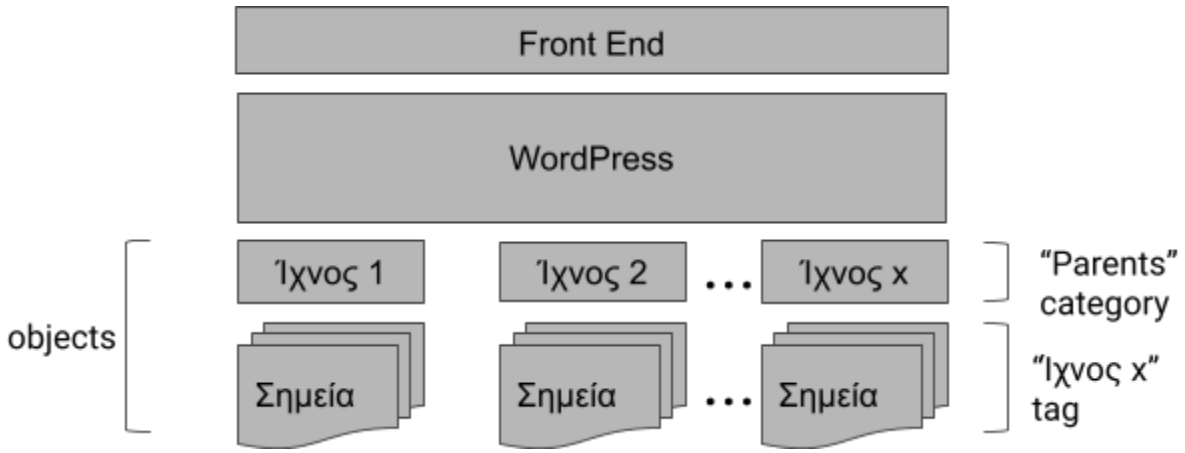
Την προβολή της μηκοτομής ενός ίχνους, την επιτυγχάνουμε και πάλι με έτοιμο shortcode, που όμως, το βάζουμε στο post του ίχνους ενδιαφέροντος.

Προβολή pois

Για την προβολή pois ενός ίχνους, ο χρήστης χρησιμοποιεί το shortcode μηκοτομής. Δηλαδή, ένα shortcode παρέχει τις δύο δυνατότητες. Ο χρήστης λοιπόν, βάζει τις ακτίνες που τον ενδιαφέρουν, στις κατηγορίες που τον ενδιαφέρουν, σε σχετική φόρμα HTML, υποβάλει τις επιλογές του, και λαμβάνει τα pois.

Οργάνωση εφαρμογής

Την οργάνωση της εφαρμογής, παρουσιάζει η Εικόνα 9 :



Εικόνα 9: Η οργάνωση της εφαρμογής.

Στο ανώτερο επίπεδο υπάρχει η επιφάνεια χρήστη (front end). Την ύπαρξη της επιφάνειας, καθώς και την ανάδραση του χρήστη με αυτήν, εξασφαλίζει η πλατφόρμα του WordPress, η οποία εκτελεί τις λειτουργίες. Κύριες γεωμετρικές οντότητες είναι τα ίχνη και τα σημεία τους.
Αναφέρουμε ότι στην εφαρμογή μας χρησιμοποιούμε τα εξής categories και tags:

- Το parents category, το οποίο περιλαμβάνει τα posts που αντιστοιχούν στα ίχνη, και όχι τα posts που αντιστοιχούν στα σημεία των ιχνών. Το σύμβολο του κάθε ίχνους είναι σημείο που ο χρήστης επιλέγει αυθαίρετα μεν, κοντά στο κέντρο βάρου δε.

- Ένα category ονόματος της μορφής “Track x”, για κάθε ίχνος x. Το category “Track x”, περιλαμβάνει το post που αντιστοιχεί στο σημειακό σύμβολο του ίχνους, και τα posts που αντιστοιχούν στα σημεία του ίχνους.
- Ένα tag της μορφής “point of track x”, για κάθε ίχνος. Προσαρτούμε το tag “point of track x” σε κάθε post σημείου του track x, ώστε να δηλώσουμε το ίχνος στο οποίο ανήκει το σημείο. Το category “Track x”, σημασιολογικά αρκεί, όμως δεν αρκεί για το λογισμικό, καθώς ο χάρτης κάθε ίχνους, τον οποίο παρέχει το Progress Map plugin, απαιτεί tags.

Σχετικά με τις σχέσεις μεταξύ posts, tags, categories, σημείων, και ιχνών, που δεν αναφέραμε στους τρεις, ανωτέρω ορισμούς:

- Κάθε ίχνος αντιστοιχεί σε ένα post. Ισχύει και το αντίστροφο, ως απαίτηση λειτουργίας της εφαρμογής.
- Κάθε ίχνος μπορεί να αντιστοιχεί σε πολλά σημεία. Επίσης, κάθε σημείο μπορεί να ανήκει σε πολλά ίχνη.
- Κάθε ίχνος μπορεί να ανήκει μόνο σε ένα category της μορφής “Track x”.

Υλοποίηση της εφαρμογής

Τις λειτουργικότητες της εφαρμογής, τις οποίες υλοποιήσαμε χωρίς προγραμματισμό, τις καλύπτουμε στα κεφάλαια "Μεθοδολογία της εφαρμογής", και "Χρήση της εφαρμογής". Στο τρέχον κεφάλαιο, σχολιάζουμε τον κώδικα, ανά λειτουργικότητα που απαιτεί κώδικα:

- Απάντηση ερωτημάτων εγγύτητας
- Ανανέωση τιμής θερμοκρασίας
- Υπολογισμός μέσης κλίσης
- Αναζήτηση ίχνους με κριτήρια εγγύτητας, θερμοκρασίας, και μέσης κλίσης
- Προβολή μηκοτομής ίχνους
- Προβολή pois

Επίσης, στο σημείο αυτό καταγράφουμε ότι στα προγράμματά μας, αρκετά μεγάλη έκταση καταλαμβάνουν σχόλια επεξήγησης, και σχόλια-κώδικας. Ως σχόλια-κώδικας, εννοούμε κυρίως τον κώδικα που χρησιμοποιήσαμε ως έλεγχο του προγράμματος, και μετά τον μετατρέψαμε σε σχόλια, ώστε να τον απενεργοποιήσουμε.

Απάντηση ερωτημάτων εγγύτητας [24]

Για μία εκ των κατηγοριών εγγύτητας, αυτήν του φαγητού, αναφέρουμε τα κύρια σημεία του αντίστοιχου προγράμματος, ώστε να μεταφέρουμε την γενική ιδέα, στον αναγνώστη. Η ιδέα αυτή παραμένει, για κάθε κατηγορία, με μικρές, αμελητέες αλλαγές.

- Λαμβάνουμε την επιθυμητή, μέγιστη απόσταση (όχι μέγιστη επιθυμητή), από τον χρήστη, μέσω φόρμας HTML.
- Λαμβάνουμε τα id όλων των ιχνών που διαθέτει η εφαρμογή μας. Τα id αυτά τα λαμβάνουμε ώστε να μπορέσουμε στην συνέχεια, να εκχωρήσουμε στο πεδίο "food" κάθε ίχνους, αλφαριθμητικό που να δηλώνει αν υπάρχει τουλάχιστον ένα σημείο "food", εντός της επιθυμητής, μέγιστης απόστασης, ή όχι.
- Για κάθε ίχνος
    - λαμβάνουμε τα id των σημείων του.
    - Χρησιμοποιούμε τα id των σημείων του ίχνους, ώστε να υπολογίσουμε το κέντρο βάρους του ίχνους.
    - Για κάθε type της κατηγορίας "φαγητό"
        - Εξετάζουμε αν υπάρχει τουλάχιστον ένα σημείο, εντός της επιθυμητής, μέγιστης απόστασης, με κέντρο το κέντρο βάρους του ίχνους.
        - Αν ναι, τότε βάζουμε αντίστοιχη τιμή στο πεδίο "food", του ίχνους.
        - Αν όχι, ομοίως.

[24] Ο πλήρης κώδικας για κάθε κατηγορία εγγύτητας, υπάρχει στο "Παράρτημα Β': Απάντηση ερωτημάτων εγγύτητας".

Υπολογισμός μέσης κλίσης [25]

- Τοποθετούμε στον κώδικα, την συνάρτηση υπολογισμού απόστασης, κατά Vincenty. Με την συνάρτηση αυτή, μπορούμε στην συνέχεια να υπολογίσουμε αποστάσεις, ώστε να βρούμε τις κλίσεις των στοιχειωδών ευθυγράμμων τμημάτων του εκάστοτε ίχνους.
- Λαμβάνουμε τα id όλων των ιχνών που διαθέτει η εφαρμογή μας, ώστε στην συνέχεια, σε δομή επανάληψης, να μπορέσουμε να υπολογίσουμε την μέση κλίση κάθε ίχνους.
- Για κάθε ίχνος
    - Λαμβάνουμε τα id των σημείων του, ώστε στην συνέχεια, να μπορέσουμε να λάβουμε το υψόμετρο καθενός εξ' αυτών.
    - Για κάθε σημείο του ίχνους
        - Λαμβάνουμε το υψόμετρο του, καθώς και το υψόμετρο του αμέσως επόμενου σημείου, με το οποίο δηλαδή σχηματίζουν στοιχειώδες, ευθύγραμμο τμήματα.
        - Υπολογίζουμε την απόσταση των δύο, κατά Vincenty.
        - Εκχωρούμε την κλίση του στοιχειώδους, ευθύγραμμα τμήματος σε άθροισμα των κλίσεων αυτών, ώστε, όταν βγούμε από την τρέχουσα δομή επανάληψης (την "Για κάθε σημείο του ίχνους", και όχι την "Για κάθε ίχνος"), να μπορέσουμε να διαιρέσουμε το άθροισμα αυτό, με τον αριθμό των στοιχειωδών ευθυγράμμων τμημάτων, και έτσι να λάβουμε την μέση κλίση του ίχνους.
    - Διαιρούμε το άθροισμα, και λαμβάνουμε την μέση κλίση του ίχνους.
    - Εκχωρούμε την μέση κλίση στο πεδίο "slope", του ίχνους, χρησιμοποιώντας το id του ίχνους, που λάβαμε παραπάνω.

Ανανέωση θερμοκρασίας [26]

- Λαμβάνουμε τα id των ιχνών, ώστε να μπορέσουμε στη συνέχεια, σε δομή επανάληψης, να εκχωρήσουμε την θερμοκρασία στο πεδίο "temp", του κάθε ίχνους.
- Για κάθε ίχνος
    - Λαμβάνουμε την θερμοκρασία
    - Εκχωρούμε την θερμοκρασία

Προβολή μηκοτομής ίχνους και εμφάνιση pois [27]

Η προβολή μηκοτομής ίχνους και η εμφάνιση pois, έχουν κάποια κοινά στοιχεία ως διαδικασίες. Έτσι, τις έχουμε εντάξει σε κοινό πρόγραμμα, γι' αυτό και τις εξηγούμε σε ένα αντί δύο εδαφίων.

---

[25] Ο πλήρης κώδικας του υπολογισμού μέσης κλίσης, υπάρχει στο "Παράρτημα Γ' : Υπολογισμός μέσης κλίσης".
[26] Ο πλήρης κώδικας ανανέωσης θερμοκρασίας, υπάρχει στο "Ανανέωση θερμοκρασίας".
[27] Ο πλήρης κώδικας προβολής μηκοτομής ίχνους και εμφάνιση pois, υπάρχει στο "Παράρτημα Ε' : Προβολή μηκοτομής ίχνους".

Σημειώνουμε επίσης, ότι ο σκελετός του τμήματος του προγράμματος, το οποίο αφορά την προβολή μηκοτομής, προκύπτει από το "Παράρτημα Α' : δείγμα "elevation along path".".

- Ο χρήστης δίνει σε φόρμα HTML, τα κριτήρια και τις αντίστοιχες ακτίνες.
- Λαμβάνουμε τα id των σημείων του ίχνους στο οποίο έχουμε βάλει το shortcode.
- Παίρνουμε τις συντεταγμένες των σημείων του ίχνους, χρησιμοποιώντας τα id που λάβαμε.
- Υπολογίζουμε το κέντρο βάρους του ίχνους, με τις παραπάνω συντεταγμένες.
- Για κάθε κατηγορία
    - Για κάθε Google type της κατηγορίας
        - Παίρνουμε τις συντεταγμένες σημείων εντός της αντίστοιχης ακτίνας που έδωσε ο χρήστης, αν υπάρχει τουλάχιστον ένα τέτοιο σημείο
    - Φτιάχνουμε μία αντίστοιχη εγγραφή, στο υπόμνημα που θα προβάλουμε, ως εξήγηση των pois στον χάρτη.
    - Προβάλουμε τα pois στον χάρτη.
- Προβάλουμε την μηκοτομή του ίχνους.

Χρήση της εφαρμογής

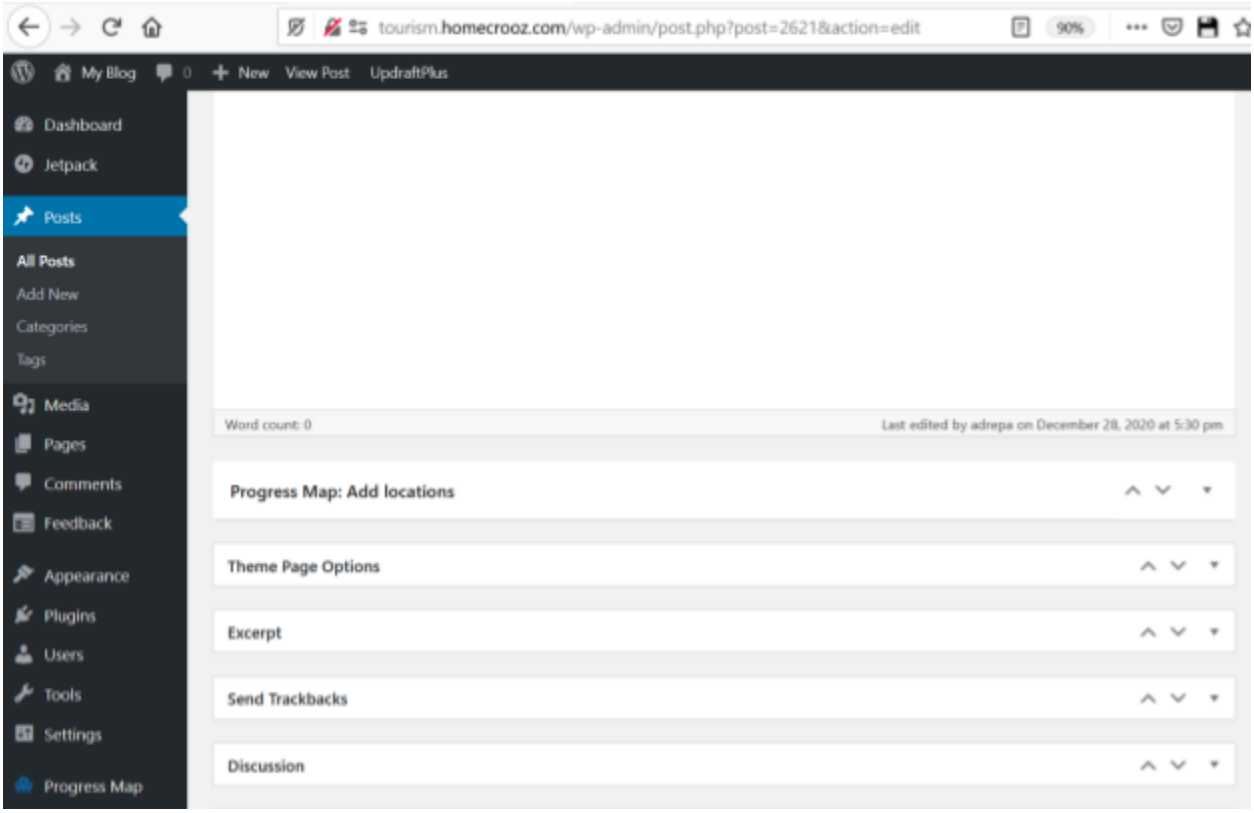
Στο παρόν κεφάλαιο, αναφέρουμε σενάρια χρήσης της εφαρμογής, καλύπτοντας τις κύριες δυνατότητες της εφαρμογής. Προκειμένου ο χρήστης να μπορέσει να αξιοποιήσει τις δυνατότητες αυτές, πρέπει να διαθέτει αρκετά WordPress capabilities. Σημειώνουμε ότι το editor role δεν επαρκεί, το οποίο, στην ιεραρχία των roles είναι ακριβώς κάτω από τον administrator. Μία επιλογή του administrator της εφαρμογής, είναι να αποδώσει στους χρήστες στους οποίους επιθυμεί να παρέξει πλήρεις δυνατότητες, το administrator role. Περιγράφουμε λοιπόν την εφαρμογή, μέσα από την οπτική του administrator. Η ανάθεση των WordPress roles γίνεται από το μενού “Users”, του dashboard.

Δημιουργία και εμφάνιση ίχνους

Έστω ότι ένας χρήστης επιθυμεί να δημιουργήσει ένα ίχνος, ως οντότητα δηλαδή, την οποία να αναγνωρίζει το WordPress, και να το εμφανίσει σε χάρτη που δείχνει μόνο το συγκεκριμένο ίχνος, και σε χάρτη που δείχνει όλα τα ίχνη, με τη μορφή σημείων-συμβόλων.

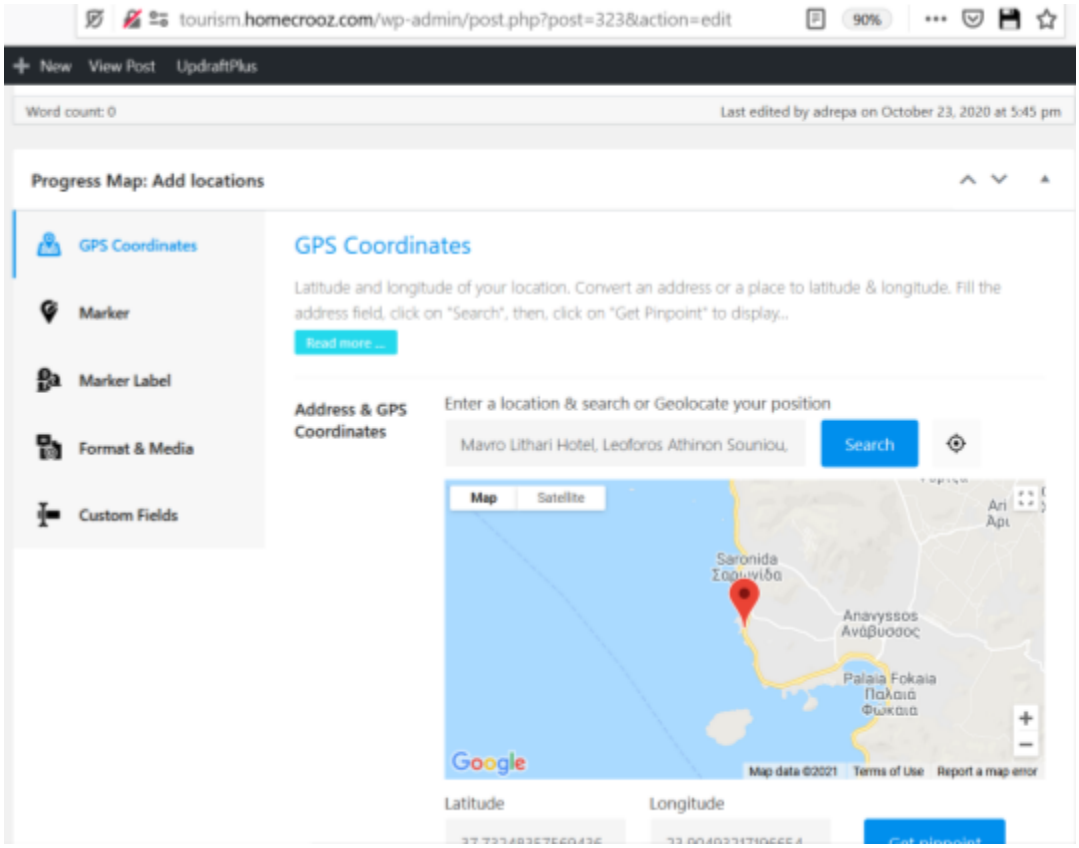
Δημιουργία ενός σημείου του ίχνους

Προκειμένου ο χρήστης να δημιουργήσει ένα ίχνος, θα πρέπει να δημιουργήσει ένα post, για κάθε σημείο του ίχνους. Εφόσον έχουμε ενεργοποιήσει ως administrator, το plugin “Progress Map”, ο χρήστης βλέπει την καρτέλα “Progress Map: Add Locations”, στην σελίδα “Edit Post”, σύμφωνα με την Εικόνα 10:



Εικόνα 10: Η καρτέλα “Progress Map: Add locations”, στην σελίδα “Edit Post”, του post που αντιστοιχεί στο σημείο που δημιουργεί ο χρήστης.

Αναπτύσσει την καρτέλα, και στο “GPS Coordinates” *, πληκτρολογεί την διεύθυνση του σημείου του ίχνους, όπως στην Εικόνα 11:

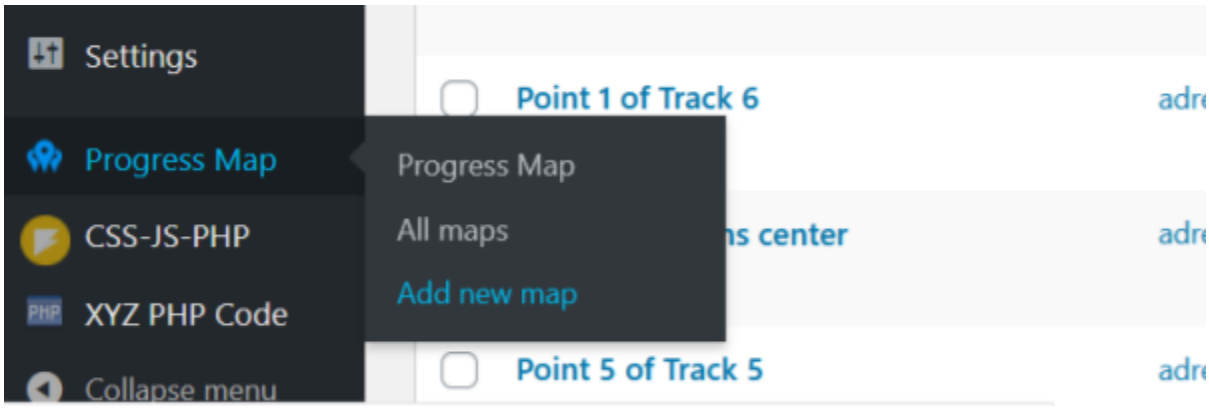


Εικόνα 11: Το μενού “GPS Coordinates”, στην καρτέλα “Progress Map: Add Locations”.

Πατά το κουμπί “Get pinpoint”, ώστε να λάβει τις συντεταγμένες του σημείου. Αν θέλει να τροποποιήσει την θέση, μπορεί να σύρει την πινέζα επί του χάρτη. Όμως, θα πρέπει να πατήσει πάλι το κουμπί “Get pinpoint”.
Στο μενού “Categories”, στο πάνω-δεξιά μέρος της σελίδας, ο χρήστης δημιουργεί μία νέα κατηγορία, πατώντας το κουμπί “+Add New Category”. Η κατηγορία πρέπει να είναι της μορφής “Track x”, όπου x, ο αριθμός του ίχνους στο οποίο ανήκει το τρέχον σημείο. Το κουτάκι δίπλα στην κατηγορία “Track x”, πρέπει να περιλαμβάνει τικ, ώστε το σημείο να αντιστοιχηθεί στο ίχνος. Ο χρήστης, δημιουργεί την κατηγορία ανά ίχνος, μόνο μία φορά, όμως αντιστοιχεί το κάθε post-σημείο του ίχνους, στην αντίστοιχη κατηγορία.
Στο μενού “Tags”, ακριβώς από κάτω, ο χρήστης ακολουθεί αντίστοιχη διαδικασία, φτιάχνοντας tag της μορφής “Point of Track x”.
Στο μενού “Featured Image”, ο χρήστης επιλέγει, εφόσον το επιθυμεί, φωτογραφία που θα μπει στο post.
Στο τελευταίο βήμα της δημιουργίας σημείου του ίχνους, στο μενού “Publish”, ο χρήστης πατά το κουμπί δημοσιοποίησης.
Εδώ σημειώνουμε, ότι, προκειμένου ο χρήστης να προσαρτήσει στο σημείο, βίντεο ή εικόνα, πατά το κουμπί “Add Media”, ακριβώς πάνω από την επιφάνεια συγγραφής κειμένου, στην σελίδα “Edit Post”.

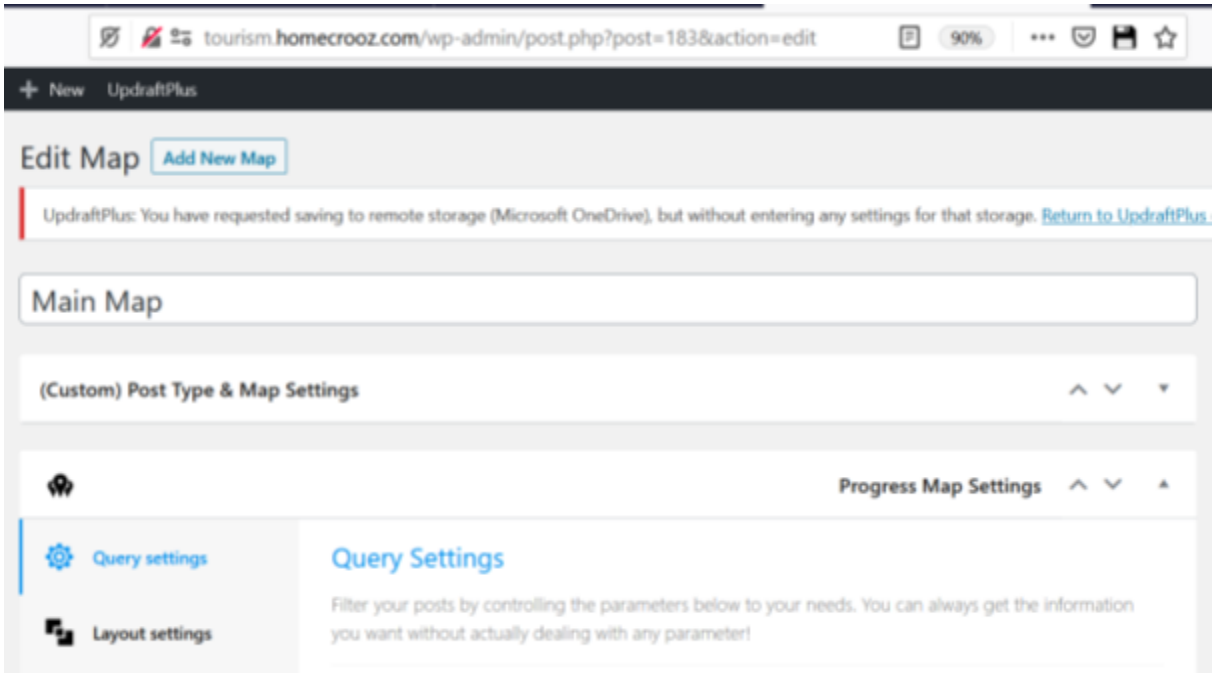
Εμφάνιση ίχνους

Αφού ο χρήστης έχει δημιουργήσει τουλάχιστον τρία σημεία για το ίχνος, δημιουργεί μία τεθλασμένη γραμμή που καθορίζει την διαδρομή. Πρώτα όμως, φτιάχνει έναν χάρτη που δείχνει τα σημεία, μόνο του τρέχοντος ίχνους.
Αυτό το επιτυγχάνει από το μενού “Progress Map”, στο dashboard του WordPress, όπου επιλέγει “Add new map”, όπως στην Εικόνα 12:

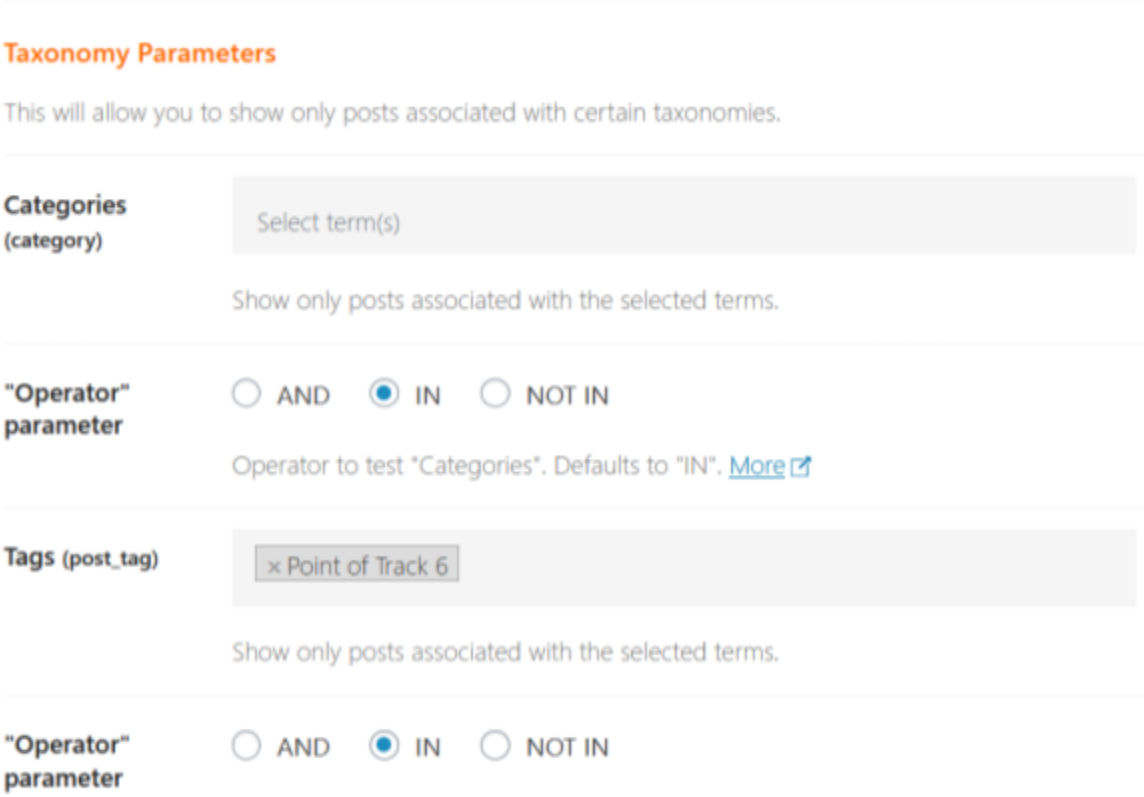


Εικόνα 12: Πού βρίσκουμε την επιλογή “Add new map”.

Ονομάζει τον χάρτη και επιλέγει ως post type, το post και όχι το page. Στο μενού “Progress Map Settings”, στην καρτέλα “Query Settings” (Εικόνα 13), στην περιοχή “Taxonomy Parameters”, επιλέγει το tag (Εικόνα 14):



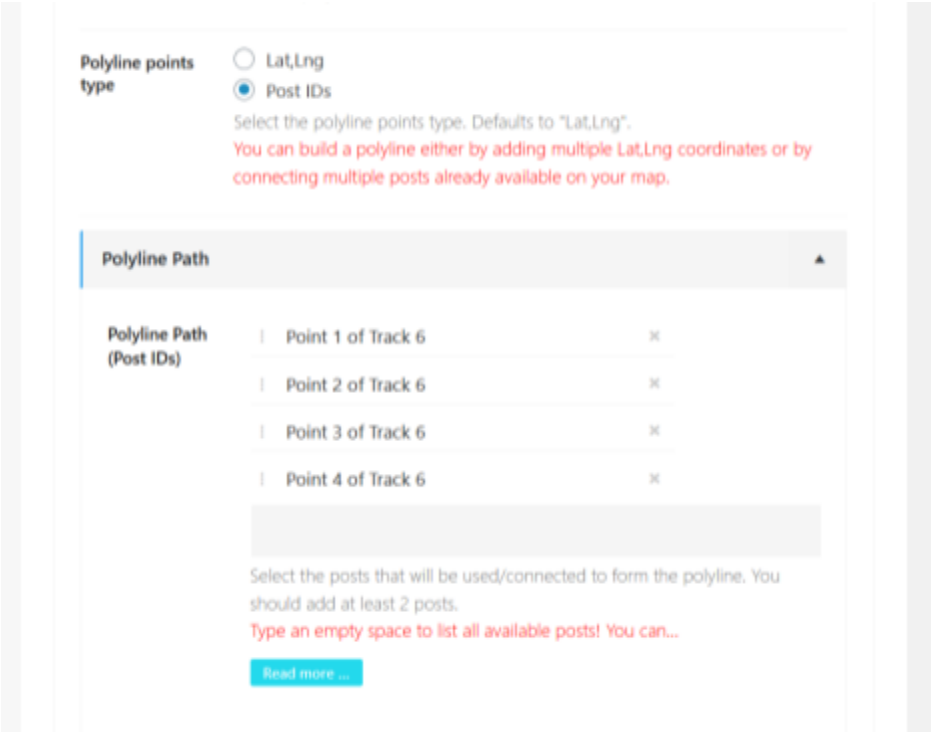
Εικόνα 13: Η καρτέλα “Query settings”.



Εικόνα 14: Η περιοχή “Taxonomy Parameters”, και το πεδίο επιλογής tag.
Με τον τρόπο αυτό, εξασφαλίζει ότι ο χάρτης αυτός δείχνει μόνο σημεία με το επιλεγμένο tag.
Στην καρτέλα “Polylines Settings”, ενεργοποιεί την επιλογή προβολής polyline, όπως στην Εικόνα 15:

Εικόνα 15: Ενεργοποίηση δυνατότητας προβολής polyline.

Στην περιοχή “Polylines”, στο μενού “Polyline Path”, καθορίζει την διαδοχή ένωσης των σημείων του ίχνους, όπως στην Εικόνα 16:



Εικόνα 16: Καθορισμός διαδοχής ένωσης σημείων ίχνους.

Ο χρήστης πατά το κουμπί “Publish”, ώστε να δημιουργήσει τον χάρτη.
Προκειμένου να βάλει τον χάρτη σε post, αντιγράφει το shortcode του χάρτη, από την επιλογή “All maps”, στο dashboard *, και το επικολλά στην επιφάνεια κειμένου. Το post αυτό, αντιστοιχεί στο συγκεκριμένο ίχνος, και σε κανένα άλλο. Μάλιστα, ο χρήστης θα πρέπει να εντάξει το post αυτό στην κατηγορία “Track x”, και στην κατηγορία “parents”, ώστε να καταστούν εφικτές διάφορες λειτουργικότητες της εφαρμογής (ζητήματα προγραμματισμού).

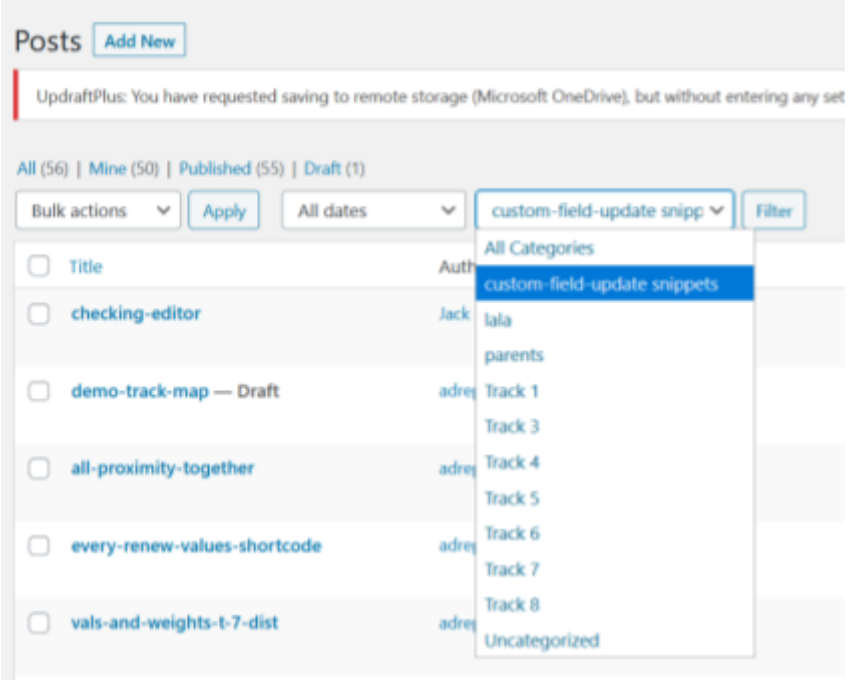
Δημιουργία και εμφάνιση σημείου-συμβόλου ίχνους

Στα αμέσως παραπάνω εδάφια, περιγράψαμε την εμφάνιση των σημείων του ίχνους, σε χάρτη που αντιστοιχεί στο ίχνος. Προκειμένου να βάλει το σημείο-σύμβολο σε χάρτη με όλα τα ίχνη-σημεία, ο χρήστης πρέπει καταρχάς να δημιουργήσει το αντικείμενο του χάρτη αυτού, ο οποίος διαφέρει από τον χάρτη που αναφέρουμε στο “Εμφάνιση ίχνους”. Δημιουργεί λοιπόν τον χάρτη με αντίστοιχο τρόπο με αυτόν που περιγράφουμε στο “Εμφάνιση ίχνους”, με την διαφορά ότι, στο μενού “Progress Map Settings”, στην καρτέλα “Query Settings”, στην περιοχή “Taxonomy Parameters”, ως operator parameter, για τα tags, και όχι για τα categories, επιλέγει το “NOT IN”. Με τον τρόπο αυτό, τα tags που επιλέγει, τα αποκλείει από τον χάρτη. Έτσι, αν ένα post-σημείο έχει tag “x”, τότε το σημείο αυτό δεν μπαίνει στον χάρτη. Δεδομένου ότι όταν φτιάχνουμε ένα σημείο ίχνους (βλέπε “Εμφάνιση ίχνους”), και όχι σημείο-σύμβολο ίχνους, προσαρτούμε σε αυτό κάποιο tag, αν συμπεριλάβουμε το tag αυτό στον χάρτη, τότε δείχνουμε μόνο το σημείο-σύμβολο ίχνους, και όχι τα σημεία που αποτελούν το ίχνος, που είναι και το επιθυμητό.

Εφόσον ο χρήστης δημιουργήσει το αντικείμενο κυρίου χάρτη, βάζει το shortcode του σε post (βλέπε “Εμφάνιση ίχνους”, για την διαδικασία), το οποίο δείχνει τα ίχνη, ως σημεία.

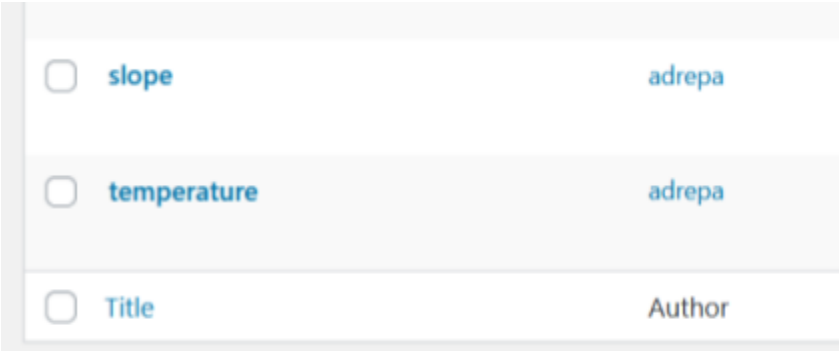
Θερμοκρασία και κλίση ίχνους

Έστω ότι ο χρήστης επιθυμεί να δει την θερμοκρασία ή την μέση κλίση ενός ίχνους. Πηγαίνει στην σελίδα με τα posts, επιλέγει την κατηγορία “custom-field-update snippets”, και κλικάρει “Filter”, όπως στην Εικόνα 17:



Εικόνα 17: Φιλτράρισμα των posts, ώστε να εμφανιστούν τα posts απάντησης ερωτημάτων, και υπολογισμού τιμών θερμοκρασίας και κλίσης.

Έτσι, προκύπτει η Εικόνα 18, με τα posts απάντησης διάφορων ερωτημάτων, και υπολογισμού τιμών θερμοκρασίας και κλίσης:



Εικόνα 18: Τα δύο post ανανέωσης θερμοκρασίας όλων των ιχνών, και υπολογισμού της μέσης κλίσης όλων των ιχνών.

Αν ενδιαφέρει τον χρήστη η θερμοκρασία κάποιου συγκεκριμένου ίχνους, επιλέγει “View”, στο post “temperature”. Έτσι, ανανεώνεται η τιμή θερμοκρασίας σε κάθε ίχνος, και συνεπώς και στο ίχνος ενδιαφέροντος. Κατόπιν, πηγαίνει στο post του ίχνους ενδιαφέροντος, και συγκεκριμένα στο “Edit post”, όπου μπορεί να δει την θερμοκρασία, ανοίγοντας το tab “Custom fields”, και κοιτώντας την τιμή του πεδίου “temp”.
Αντίστοιχη είναι η διαδικασία για την κλίση ίχνους. Βέβαια, η περίπτωση αυτή δεν έχει την χρησιμότητα ανανέωσης, αλλά του υπολογισμού της κλίσης νέων ιχνών.

Λήψη ιχνών που ικανοποιούν κάποια κριτήρια

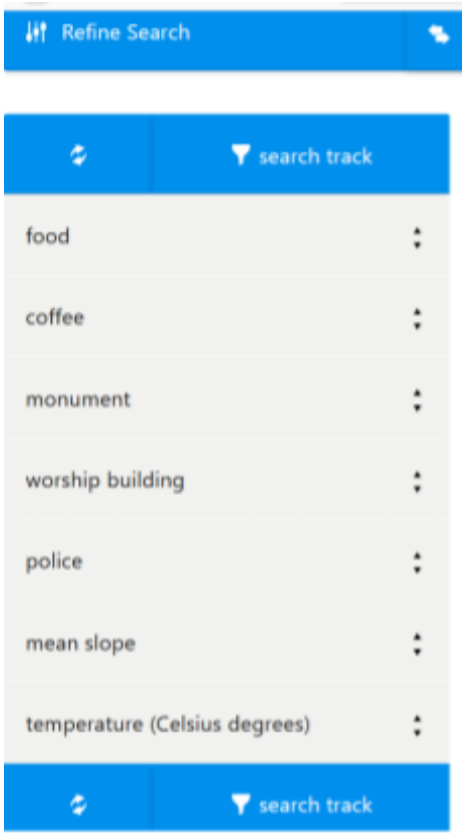
Έστω ότι ο χρήστης επιθυμεί να δει ίχνη τα οποία ικανοποιούν κάποια κριτήρια. Τα διαθέσιμα κριτήρια είναι:

- Εγγύτητας
    - Φαγητού: bakery, meal takeaway, restaurant
    - Αστυνομίας: police
    - Καφετέριας: bar, cafe
    - Σημείου ενδιαφέροντος για τουρίστες: tourist attraction
    - Θρησκευτικού κτιρίου: church, hindu temple, mosque, synagogue
- Θερμοκρασίας και κλίσης

Καταρχάς, ο χρήστης ανανεώνει τις τιμές θερμοκρασίας και κλίσης, εφόσον οι τιμές αυτές τον ενδιαφέρουν, όπως περιγράφουμε στο “Θερμοκρασία και κλίση ίχνους”. Επίσης, δίνει εντολή στο σύστημα, να μάθει εάν υπάρχει τουλάχιστον ένα αντικείμενο της κάθε κατηγορίας, στις επιθυμητές αποστάσεις. Αυτό το κάνει, προβάλλοντας τα σχετικά posts, όπως δείχνουμε στην Εικόνα 17. Για παράδειγμα, αν τον ενδιαφέρει η ύπαρξη τουλάχιστον ενός αντικειμένου της κατηγορίας “Φαγητό”, εντός ακτίνας χιλίων μέτρων, από το κέντρο βάρους του ίχνους, επιλέγει το post “food”, με αποτέλεσμα την Εικόνα 19:

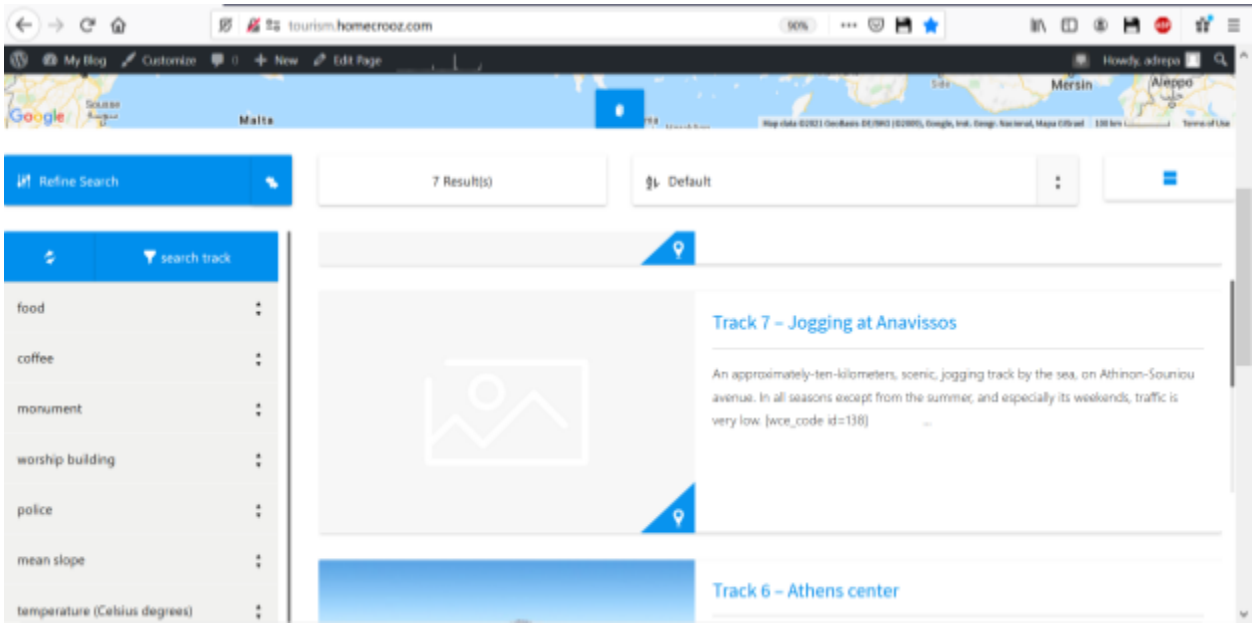
Εικόνα 19: Υποβολή απόστασης.

Στην συνέχεια, πηγαίνει στην κεντρική σελίδα του ιστοτόπου, και συγκεκριμένα, στην λίστα με τα φίλτρα, την οποία βλέπουμε στην Εικόνα 20:



Εικόνα 20: Η λίστα επιλογής φίλτρων, και αντίστοιχων τιμών, για την αναζήτηση ιχνών.

Αναπτύσσει τις αντίστοιχες καρτέλες, επιλέγει 1, ώστε να ενεργοποιήσει το κριτήριο αναζήτησης, θέτει τα εύρη κλίσης και θερμοκρασίας, αν τον ενδιαφέρουν, και πατά “search track”. Τα ίχνη που πληρούν τα κριτήρια, προκύπτουν ακριβώς δεξιά, όπως στην Εικόνα 21:

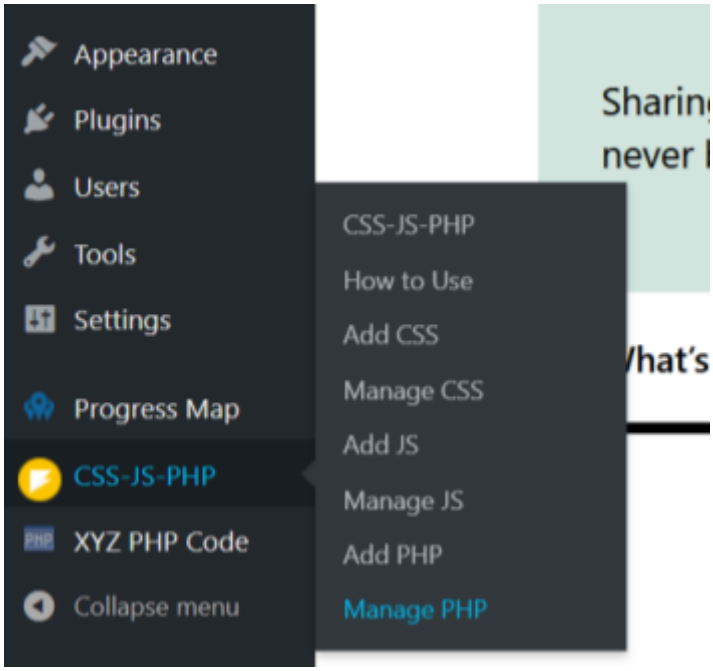


Εικόνα 21: Τα ίχνη που ικανοποιούν τα κριτήρια, προκύπτουν στα δεξιά της λίστας αναζήτησης.

Αν ο χρήστης πατήσει, για παράδειγμα το ίχνος 6, το αντίστοιχο post εμφανίζεται στην ίδια σελίδα, αλλά σε ένθετο παράθυρο.

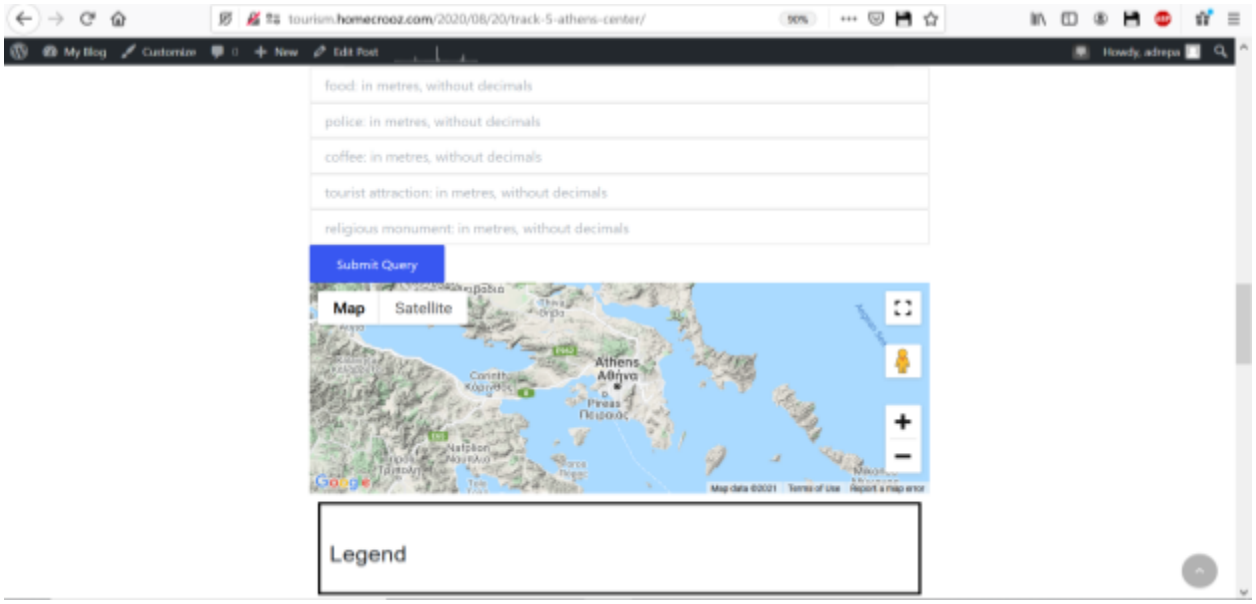
Εμφάνιση σημείων ενδιαφέροντος (pois)

Έστω ότι ο χρήστης επιθυμεί να δει, σε χάρτη, τα σημεία ενδιαφέροντος, κατά Google Maps πάντα, γύρω από κάποιο ίχνος. Αν ο δημιουργός του post, για το ίχνος που ενδιαφέρει τον χρήστη, δεν έχει παρέξει την δυνατότητα αυτή για τον χρήστη της εφαρμογής, τότε πρέπει να το κάνει ο χρήστης. Δηλαδή, από το dashboard του WordPress, ανοίγει το μενού του plugin “CSS-JS-PHP”, και πατά “Manage PHP”, όπως στην Εικόνα 22:



Εικόνα 22: Άνοιγμα σελίδας “Manage PHP”, του plugin “CSS-JS-PHP”.

Στην σελίδα που ανοίγει, ο χρήστης βρίσκει το shortcode εμφάνισης χάρτη με pois, δηλαδή το [wce_code id=150], με τίτλο “show-pois-and-elev-along”. Το shortcode αυτό, το επικολλά στο post του ίχνους που τον ενδιαφέρει, και ανανεώνει το post. Πλέον, βλέπει την Εικόνα 23:



Εικόνα 23: Φόρμα υποβολής αποστάσεων, για διάφορες κατηγορίες, όπως “Φαγητό”, και ο αντίστοιχος χάρτης με το υπόμνημα.

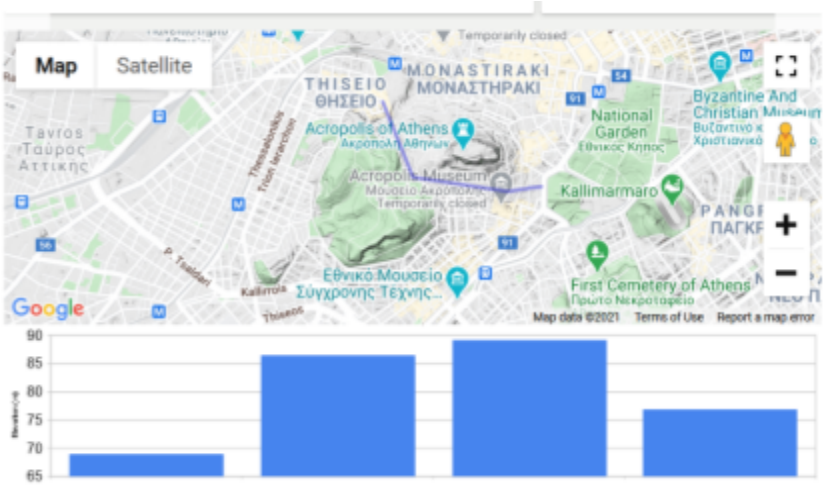
Ο χρήστης βάζει όσες αποστάσεις επιθυμεί, και προκύπτει, για παράδειγμα, η Εικόνα 24, για απόσταση χιλίων μέτρων για “Φαγητό”, και χιλίων μέτρων για “Σημείο Τουριστικού Ενδιαφέροντος”:

Εικόνα 24: Προβολή pois.

Ο χάρτης μπορεί να μεγεθύνει, να κάνει pan, να μεταβεί σε “Google Street View”, και να δει τον χάρτη σε πλήρη οθόνη.

Εμφάνιση μηκοτομής

Έστω ότι ο χρήστης επιθυμεί να προβάλει την μηκοτομή κάποιου ίχνους. Αντίστοιχα με το “Εμφάνιση σημείων ενδιαφέροντος (pois)”, πιθανώς θα χρειαστεί να βάλει το κατάλληλο shortcode, [wce_code id=95], στο post, στην επιφάνεια συγγραφής κειμένου. Όταν ανανεώσει το post, βλέπει την Εικόνα 25:



Εικόνα 25: Μηκοτομή ίχνους

Αν ο χρήστης το επιθυμεί, πατώντας σε κάποια από τις μπλε μπάρες, λαμβάνει την τιμή υψομέτρου σε συγκεκριμένη κορυφή του ίχνους.

Αξιολόγηση της εφαρμογής, και προτάσεις τροποποίησης/βελτίωσης.

Θετικά στοιχεία της εφαρμογής

Ένα θετικό στοιχείο της εφαρμογής είναι ότι αξιοποιεί την χωρική φύση του προβλήματος επιλογής προορισμού. Αυτό το επιτυγχάνει, κυρίως μέσω της ενσωμάτωσης της έννοιας της εγγύτητας, καθώς και αντίστοιχων χαρτών.
Άλλο θετικό στοιχείο της εφαρμογής είναι ότι δίνει εξέχουσα θέση στην πληροφορία που μπορεί να παρέξει οποιοσδήποτε γνωρίζει μία περιοχή (VGI), αφού ο χρήστης με επαρκή δικαιώματα χρήσης της εφαρμογής, μπορεί να την εμπλουτίσει με σχόλια, στην απλούστερη περίπτωση, αλλά και με επισήμανση διαδρομών σε χάρτη, στην πιο σύνθετη.

Αρνητικά στοιχεία της εφαρμογής

Ένα αρνητικό στοιχείο της εφαρμογής, είναι η πολυπλοκότητα αξιοποίησης ορισμένων δυνατοτήτων της, όπως εύκολα διαπιστώνει κανείς, από το “Χρήση της εφαρμογής”.
Άλλο αρνητικό στοιχείο είναι η ανάγκη ελέγχου της ποιότητας της VGI, αν επιθυμούμε να παρέχουμε αξιοπιστία.
Ακόμη, αρνητικό είναι ότι χρησιμοποιούμε ευκλείδεια αντί δικτυακής απόστασης. Συνεπώς, όταν δηλώνουμε ότι ένα αντικείμενο είναι κοντά σε κάποιο ίχνος, αυτό μπορεί και να μην ισχύει, και η αδυναμία ίσως είναι ανάλογη της απόκλισης της ευθείας απόστασης, από την δικτυακή. Άρα, δίνουμε μία εκτίμηση, της οποίας η ποιότητα ποικίλλει, και δεν ξέρουμε και πόσο.

Προτάσεις τροποποίησης

Με βάση τα αρνητικά στοιχεία της εφαρμογής, προτείνουμε απλοποίηση των διαδικασιών αξιοποίησης των δυνατοτήτων της εφαρμογής. Αυτό, κατά πάσα πιθανότητα απαιτεί τροποποίηση του υπάρχοντος κώδικα. Επιπλέον, προτείνουμε χρήση δικτυακής απόστασης, πιθανώς, μέσω αξιοποίησης των αποστάσεων που διαθέτουν τα Google-Maps APIs.

Βιβλιογραφία

https://www.unwto.org/why-tourism

Batty, Michael, Andrew Hudson-Smith, Richard Milton, and Andrew Crooks. “Map Mashups, Web 2.0 and the GIS Revolution.” *Annals of GIS* 16, no. 1 (April 22, 2010): 1–13. https://doi.org/10.1080/19475681003700831.

Brownsword, Lisa L., David J. Carney, David Fisher, Grace Lewis, and Craig Meyers. “Current Perspectives on Interoperability:” Fort Belvoir, VA: Defense Technical Information Center, March 1, 2004. https://doi.org/10.21236/ADA421613.

“Cascading Style Sheets.” Accessed November 11, 2020. https://www.w3.org/Style/CSS/Overview.en.html.

“Content Management System.” In *Wikipedia*, October 7, 2020. https://en.wikipedia.org/w/index.php?title=Content_management_system&oldid=982253727.

El-Ashmawy, Khalid L.A. “Investigation of the Accuracy of Google Earth Elevation Data.” *Artificial Satellites* 51, no. 3 (September 1, 2016): 89–97. https://doi.org/10.1515/arsa-2016-0008.

“Free and Open Source Software.” Accessed October 9, 2020. https://www2.cs.siu.edu/~carver/talks/foss.pdf.

“GeoJSON—ArcGIS Online Help | Documentation.” Accessed October 10, 2020. https://doc.arcgis.com/en/arcgis-online/reference/geojson.htm.

Goodchild, Michael F. “CITIZENS AS SENSORS: THE WORLD OF VOLUNTEERED GEOGRAPHY,” n.d., 15.

Google Developers. “Google Maps Platform FAQ.” Accessed October 8, 2020. https://developers.google.com/maps/faq.

MDN Web Docs. “HTML: HyperText Markup Language.” Accessed November 11, 2020. https://developer.mozilla.org/en-US/docs/Web/HTML.

MDN Web Docs. “Introduction to Web APIs.” Accessed October 8, 2020. https://developer.mozilla.org/en-US/docs/Learn/JavaScript/Client-side_web_APIs/Introduction.

“JSON.” Accessed October 10, 2020. https://www.json.org/json-en.html.

WordPress.org Forums. “Pages,” October 12, 2018. https://wordpress.org/support/article/pages/.

“PHP: Preface - Manual.” Accessed November 11, 2020. https://www.php.net/manual/en/preface.php.

WordPress.org Forums. “Taxonomies,” November 2, 2018. https://wordpress.org/support/article/taxonomies/.

“The Home of Location Technology Innovation and Collaboration | OGC.” Accessed October 9, 2020. https://www.ogc.org/.

Ubukawa, Taro. “An Evaluation of the Horizontal Positional Accuracy of Google and Bing Satellite Imagery and Three Roads Data Sets Based on High Resolution Satellite Imagery,” n.d., 16.

Vescoukis, Vassilios. “5. Αρχιτεκτονικές Εφαρμογών Διαδικτύου.” *COMPUTATIONAL METHODS*, n.d., 100.

“What Is Cloud Computing?,” August 20, 2020. https://www.ibm.com/cloud/learn/cloud-computing.

“What Is GIS? | Geographic Information System Mapping Technology.” Accessed October 9, 2020. https://www.esri.com/en-us/what-is-gis/overview.

MDN Web Docs. “What Is JavaScript?” Accessed November 11, 2020. https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First_steps/What_is_JavaScri

pt.
“What Is Open Data?” Accessed October 9, 2020.
https://opendatahandbook.org/guide/en/what-is-open-data/.
“Why Tourism? | UNWTO.” Accessed August 22, 2020. https://www.unwto.org/why-tourism.

Παράρτημα Α : δείγμα “elevation along path”.

```
<!DOCTYPE html>
<html>
  <head>
    <title>Showing Elevation Along a Path</title>
    <script
src="https://polyfill.io/v3/polyfill.min.js?features=default"></script>
    <script src="https://www.google.com/jsapi"></script>
    <script

src="https://maps.googleapis.com/maps/api/js?key=YOUR_API_KEY&callback=init
Map&libraries=&v=weekly"
      defer
    ></script>
    <style type="text/css">
      /* Always set the map height explicitly to define the size of the div
       * element that contains the map. */
      #map {
        height: 100%;
      }

      /* Optional: Makes the sample page fill the window. */
      html,
      body {
        height: 100%;
        margin: 0;
        padding: 0;
      }
    </style>
    <script>
      // Load the Visualization API and the columnchart package.
      google.load("visualization", "1", { packages: ["columnchart"] });

      function initMap() {
        // The following path marks a path from Mt. Whitney, the highest
point in the
        // continental United States to Badwater, Death Valley, the lowest
point.
        const path = [
          { lat: 36.579, lng: -118.292 },
          { lat: 36.606, lng: -118.0638 },
          { lat: 36.433, lng: -117.951 },
```

```
          { lat: 36.588, lng: -116.943 },
          { lat: 36.34, lng: -117.468 },
          { lat: 36.24, lng: -116.832 },
        ]; // Badwater, Death Valley
        const map = new google.maps.Map(document.getElementById("map"), {
          zoom: 8,
          center: path[1],
          mapTypeId: "terrain",
        });
        // Create an ElevationService.
        const elevator = new google.maps.ElevationService();
        // Draw the path, using the Visualization API and the Elevation
service.
        displayPathElevation(path, elevator, map);
      }

      function displayPathElevation(path, elevator, map) {
        // Display a polyline of the elevation path.
        new google.maps.Polyline({
          path: path,
          strokeColor: "#0000CC",
          strokeOpacity: 0.4,
          map: map,
        });
        // Create a PathElevationRequest object using this array.
        // Ask for 256 samples along that path.
        // Initiate the path request.
        elevator.getElevationAlongPath(
          {
            path: path,
            samples: 256,
          },
          plotElevation
        );
      }

      // Takes an array of ElevationResult objects, draws the path on the
map
      // and plots the elevation profile on a Visualization API
ColumnChart.
      function plotElevation(elevations, status) {
        const chartDiv = document.getElementById("elevation_chart");
```

```
        if (status !== "OK") {
          // Show the error code inside the chartDiv.
          chartDiv.innerHTML =
            "Cannot show elevation: request failed because " + status;
          return;
        }
        // Create a new chart in the elevation_chart DIV.
        const chart = new google.visualization.ColumnChart(chartDiv);
        // Extract the data from which to populate the chart.
        // Because the samples are equidistant, the 'Sample'
        // column here does double duty as distance along the
        // X axis.
        const data = new google.visualization.DataTable();
        data.addColumn("string", "Sample");
        data.addColumn("number", "Elevation");

        for (let i = 0; i < elevations.length; i++) {
          data.addRow(["", elevations[i].elevation]);
        }
        // Draw the chart using the data within its DIV.
        chart.draw(data, {
          height: 150,
          legend: "none",
          titleY: "Elevation (m)",
        });
      }
    </script>
  </head>
  <body>
    <div>
      <div id="map" style="height: 250px"></div>
      <div id="elevation_chart"></div>
    </div>
  </body>
</html>
```

Παράρτημα Β : Απάντηση ερωτημάτων εγγύτητας

1: Κατηγορία “φαγητό”

```
<!DOCTYPE html>
<html>
  <body>
    <form method="post" action="">
      <input type="text" name="maximum_distance" placeholder="in metres,
without decimals">
      <input type="submit">
    </form>
  </body>
  <?php
    // εκχωρω την επιθυμητη, μεγιστη (οχι μεγιστη, επιθυμητη) αποσταση που δινει
ο χρηστης
    $max_distance = $_POST['maximum_distance'];

    // παιρνω τα id των "parents" post
    $args = array(
      'numberposts'      => -1,
      'category'         => 25,
      'orderby'          => 'date',
      'order'            => 'DESC',
      'include'          => array(),
      'exclude'          => array(),
      'meta_key'         => '',
      'meta_value'       => '',
      'post_type'        => 'post',
      'suppress_filters' => true,
      'fields'           => 'ids'
    );
    $parents_ids_array = get_posts ( $args );

    // λαμβανω τον αριθμο των track
    $parents_ids_array_size = sizeof ( $parents_ids_array );

    // φτιαχνω array με τα food types
    $food_types = array ( "bakery", "meal_takeaway", "restaurant" );

    // για καθε track
    $n = 0;
```

```
    while ( $n <= $parents_ids_array_size - 1 ) {
      // update_metadata ( $meta_type = 'post', $object_id =
$parents_ids_array[$n], $meta_key = 'food', $meta_value = 'food_0' );

      // βρισκω το id
      $current_post_id = $parents_ids_array[$n];

      // προσοχη: argument το post id, και οχι καποιο category id
      // παιρνω τα data της αντιστοιχης κατηγοριας
      $categ_data =  get_the_category( $current_post_id );

      // υπολογιζω τα category που του αντιστοιχουν
      // προσοχη: υπολογιζω των αριθμο των category
      // και οχι τον αριθμο των track
      $categ_data_size = sizeof ( $categ_data );

      // απο τα διαφορα category στα οποια μπορει να ανηκει το ΚΑΘΕ post που
ανηκει
      // στο "parents" category, παιρνω το category με format "track x"
      $i = 0;
      while ( $i <= ( $categ_data_size - 1 ) ) {
        ${"categ_data_$i"} = $categ_data[$i];
        ${"vars_categ_data_$i"} = get_object_vars ( ${"categ_data_$i"} );
        ${"categ_$i"} = ${"vars_categ_data_$i"}[name];

        if ( strpos ( $haystack = ${"categ_$i"} , $needle = 'Track' ) !==
false ) {
          $wanted_cat = ${"categ_$i"};
          $i = 10000;
        }

        $i++;
      }

      // παιρνω το id του $wanted_cat (Track x), προσοχη, οχι το id του post,
ουτε του parents category
      $wanted_cat_id = get_cat_ID ( $wanted_cat );

      // παιρνω τα id των point(τα θελω ωστε να βρω το κεντρο βαρους τους)
      $args = array(
        'numberposts'      => -1,
        'category'         => $wanted_cat_id,
        'orderby'          => 'date',
```

```
        'order'            => 'DESC',
        'include'          => array(),
        'exclude'          => array( $parents_ids_array[$n] ),
        'meta_key'         => 'codespacing_progress_map_lat',
        'meta_value'       => '',
        'post_type'        => 'post',
        'suppress_filters' => true,
        'fields'           => 'ids'
      );
      $ids_array = get_posts ( $args );

      // ουσιαστικα, λαμβανω τον αριθμο των point
      $posts_array_size = sizeof ( $ids_array );

      // για καθε point του track
      $j = 0;
      $lats_sum = 0;
      $lngs_sum = 0;
      while ( $j <= $posts_array_size - 1 ) {
        // λαμβανω τις συντεταγμενες
        $post_lat = get_metadata ( $meta_type = 'post', $object_id =
$ids_array[$j], $meta_key = 'codespacing_progress_map_lat', $single = true
);
        $post_lng = get_metadata ( $meta_type = 'post', $object_id =
$ids_array[$j], $meta_key = 'codespacing_progress_map_lng', $single = true
);

        $lats_sum += $post_lat;
        $lngs_sum += $post_lng;

        $j++;
      }

      // υπολογιζω το κεντρο βαρους
      $cent_lat = $lats_sum/$posts_array_size;
      $cent_lng = $lngs_sum/$posts_array_size;

      // για καθε type
      $f = 0;
      while ( $f <= 3 - 1 ) {
        // λαμβανω τα point που το google βρισκει, εντος της μεγιστης,
επιθυμητης αποστασης
        $url =
```

```
"https://maps.googleapis.com/maps/api/place/nearbysearch/json?location={$ce
nt_lat},{$cent_lng}&radius={$max_distance}&type={$food_types[$f]}&key=AIzaS
yAC9YdWFl62YplTmM8J0OwxZZo0VXt_KpE";
        $html = file_get_contents ( $url );
        $html_dec = json_decode ( $html );
        $html_dec_vars = get_object_vars ( $html_dec );
        // $results = $html_dec_vars[results];
        $status = $html_dec_vars[status];
        if ( $status == 'OK') {
          update_metadata ( $meta_type = 'post', $object_id =
$parents_ids_array[$n], $meta_key = 'food', $meta_value = "at least one" );
          $f = 10000;
        } else {
          update_metadata ( $meta_type = 'post', $object_id =
$parents_ids_array[$n], $meta_key = 'food', $meta_value = "none" );
        }

        $f++;
      }

      $n++;
    }
  ?>
</html>
```

2: Κατηγορία “αστυνομία”.

```
<!DOCTYPE html>
<html>
  <body>
    <form method="post" action="">
      <input type="text" name="maximum_distance" placeholder="in metres,
without decimals">
      <input type="submit">
    </form>
  </body>
  <?php
    // εκχωρω την επιθυμητη, μεγιστη (οχι μεγιστη, επιθυμητη) αποσταση που δινει
ο χρηστης
    $max_distance = $_POST['maximum_distance'];

    // παιρνω τα id των "parents" post
```

```
    $args = array(
      'numberposts'      => -1,
      'category'         => 25,
      'orderby'          => 'date',
      'order'            => 'DESC',
      'include'          => array(),
      'exclude'          => array(),
      'meta_key'         => '',
      'meta_value'       => '',
      'post_type'        => 'post',
      'suppress_filters' => true,
      'fields'           => 'ids'
    );
    $parents_ids_array = get_posts ( $args );

    // λαμβανω τον αριθμο των track
    $parents_ids_array_size = sizeof ( $parents_ids_array );

    // για καθε track
    $n = 0;
    while ( $n <= $parents_ids_array_size - 1 ) {
      // update_metadata ( $meta_type = 'post', $object_id =
$parents_ids_array[$n], $meta_key = 'food', $meta_value = 'food_0' );

      // βρισκω το id
      $current_post_id = $parents_ids_array[$n];

      // προσοχη: argument το post id, και οχι καποιο category id
      // παιρνω τα data της αντιστοιχης κατηγοριας
      $categ_data =  get_the_category( $current_post_id );

      // υπολογιζω τα category που του αντιστοιχουν
      // προσοχη: υπολογιζω των αριθμο των category
      // και οχι τον αριθμο των track
      $categ_data_size = sizeof ( $categ_data );

      // απο τα διαφορα category στα οποια μπορει να ανηκει το ΚΑΘΕ post που
ανηκει
      // στο "parents" category, παιρνω το category με format "track x"
      $i = 0;
      while ( $i <= ( $categ_data_size - 1 ) ) {
        ${"categ_data_$i"} = $categ_data[$i];
        ${"vars_categ_data_$i"} = get_object_vars ( ${"categ_data_$i"} );
```

```
        ${"categ_$i"} = ${"vars_categ_data_$i"}[name];

        if ( strpos ( $haystack = ${"categ_$i"} , $needle = 'Track' ) !==
false ) {
          $wanted_cat = ${"categ_$i"};
          $i = 10000;
        }

        $i++;
      }

      // παιρνω το id του $wanted_cat (Track x), προσοχη, οχι το id του post,
ουτε του parents category
      $wanted_cat_id = get_cat_ID ( $wanted_cat );

      // παιρνω τα id των point(τα θελω ωστε να βρω το κεντρο βαρους τους)
      $args = array(
        'numberposts'      => -1,
        'category'         => $wanted_cat_id,
        'orderby'          => 'date',
        'order'            => 'DESC',
        'include'          => array(),
        'exclude'          => array( $parents_ids_array[$n] ),
        'meta_key'         => 'codespacing_progress_map_lat',
        'meta_value'       => '',
        'post_type'        => 'post',
        'suppress_filters' => true,
        'fields'           => 'ids'
      );
      $ids_array = get_posts ( $args );

      // ουσιαστικα, λαμβανω τον αριθμο των point
      $posts_array_size = sizeof ( $ids_array );

      // για καθε point του track
      $j = 0;
      $lats_sum = 0;
      $lngs_sum = 0;
      while ( $j <= $posts_array_size - 1 ) {
        // λαμβανω τις συντεταγμενες
        $post_lat = get_metadata ( $meta_type = 'post', $object_id =
$ids_array[$j], $meta_key = 'codespacing_progress_map_lat', $single = true
);
```

```
        $post_lng = get_metadata ( $meta_type = 'post', $object_id =
$ids_array[$j], $meta_key = 'codespacing_progress_map_lng', $single = true
);

        $lats_sum += $post_lat;
        $lngs_sum += $post_lng;

        $j++;
      }

      // υπολογιζω το κεντρο βαρους
      $cent_lat = $lats_sum/$posts_array_size;
      $cent_lng = $lngs_sum/$posts_array_size;

      // λαμβανω τα point που το google βρισκει, εντος της μεγιστης, επιθυμητης
αποστασης
      $url =
"https://maps.googleapis.com/maps/api/place/nearbysearch/json?location={$ce
nt_lat},{$cent_lng}&radius={$max_distance}&type=police&key=AIzaSyAC9YdWFl62
YplTmM8J0OwxZZo0VXt_KpE";
      $html = file_get_contents ( $url );
      $html_dec = json_decode ( $html );
      $html_dec_vars = get_object_vars ( $html_dec );
      // $results = $html_dec_vars[results];
      $status = $html_dec_vars[status];
      if ( $status == 'OK') {
        update_metadata ( $meta_type = 'post', $object_id =
$parents_ids_array[$n], $meta_key = 'police', $meta_value = "at least one"
);
        $f = 10000;
      } else {
        update_metadata ( $meta_type = 'post', $object_id =
$parents_ids_array[$n], $meta_key = 'police', $meta_value = "none" );
      }

      $n++;
    }
  ?>
</html>
```

3: Κατηγορία “Καφές”

```
<!DOCTYPE html>
<html>
  <body>
    <form method="post" action="">
      <input type="text" name="maximum_distance" placeholder="in metres,
without decimals">
      <input type="submit">
    </form>
  </body>
  <?php
    // εκχωρω την επιθυμητη, μεγιστη (οχι μεγιστη, επιθυμητη) αποσταση που δινει
ο χρηστης
    $max_distance = $_POST['maximum_distance'];

    // παιρνω τα id των "parents" post
    $args = array(
      'numberposts'      => -1,
      'category'         => 25,
      'orderby'          => 'date',
      'order'            => 'DESC',
      'include'          => array(),
      'exclude'          => array(),
      'meta_key'         => '',
      'meta_value'       => '',
      'post_type'        => 'post',
      'suppress_filters' => true,
      'fields'           => 'ids'
    );
    $parents_ids_array = get_posts ( $args );

    // print_r ( $parents_ids_array );

    // λαμβανω τον αριθμο των track
    $parents_ids_array_size = sizeof ( $parents_ids_array );

    // φτιαχνω array με τα food types
    $food_types = array ( "bar", "cafe" );

    // print_r ( $food_types );

    // για καθε track
```

```
    $n = 0;
    while ( $n <= $parents_ids_array_size - 1 ) {
      // update_metadata ( $meta_type = 'post', $object_id =
$parents_ids_array[$n], $meta_key = 'coffee', $meta_value = '0' );

      // βρισκω το id
      $current_post_id = $parents_ids_array[$n];

      // προσοχη: argument το post id, και οχι καποιο category id
      // παιρνω τα data της αντιστοιχης κατηγοριας
      $categ_data =  get_the_category( $current_post_id );

      // υπολογιζω τα category που του αντιστοιχουν
      // προσοχη: υπολογιζω των αριθμο των category
      // και οχι τον αριθμο των track
      $categ_data_size = sizeof ( $categ_data );

      // απο τα διαφορα category στα οποια μπορει να ανηκει το ΚΑΘΕ post που
ανηκει
      // στο "parents" category, παιρνω το category με format "track x"
      $i = 0;
      while ( $i <= ( $categ_data_size - 1 ) ) {
        ${"categ_data_$i"} = $categ_data[$i];
        ${"vars_categ_data_$i"} = get_object_vars ( ${"categ_data_$i"} );
        ${"categ_$i"} = ${"vars_categ_data_$i"}[name];

        if ( strpos ( $haystack = ${"categ_$i"} , $needle = 'Track' ) !==
false ) {
          $wanted_cat = ${"categ_$i"};
          $i = 10000;
        }

        $i++;
      }

      // παιρνω το id του $wanted_cat (Track x), προσοχη, οχι το id του post,
ουτε του parents category
      $wanted_cat_id = get_cat_ID ( $wanted_cat );

      // παιρνω τα id των point(τα θελω ωστε να βρω το κεντρο βαρους τους)
      $args = array(
        'numberposts'      => -1,
        'category'         => $wanted_cat_id,
```

```
        'orderby'          => 'date',
        'order'            => 'DESC',
        'include'          => array(),
        'exclude'          => array( $parents_ids_array[$n] ),
        'meta_key'         => 'codespacing_progress_map_lat',
        'meta_value'       => '',
        'post_type'        => 'post',
        'suppress_filters' => true,
        'fields'           => 'ids'
      );
      $ids_array = get_posts ( $args );

      // print_r ( $ids_array );

      // ουσιαστικα, λαμβανω τον αριθμο των point
      $posts_array_size = sizeof ( $ids_array );

      // για καθε point του track
      $j = 0;
      $lats_sum = 0;
      $lngs_sum = 0;
      while ( $j <= $posts_array_size - 1 ) {
        // λαμβανω τις συντεταγμενες
        $post_lat = get_metadata ( $meta_type = 'post', $object_id =
$ids_array[$j], $meta_key = 'codespacing_progress_map_lat', $single = true
);
        $post_lng = get_metadata ( $meta_type = 'post', $object_id =
$ids_array[$j], $meta_key = 'codespacing_progress_map_lng', $single = true
);

        $lats_sum += $post_lat;
        $lngs_sum += $post_lng;

        $j++;
      }

      // υπολογιζω το κεντρο βαρους
      $cent_lat = $lats_sum/$posts_array_size;
      $cent_lng = $lngs_sum/$posts_array_size;

      // print "$cent_lat!!!!";
      // print "$cent_lng!!!!";
```

```
      // για καθε type
      $f = 0;
      while ( $f <= 2 - 1 ) {
        // λαμβανω τα point που το google βρισκει, εντος της μεγιστης,
επιθυμητης αποστασης
        $url =
"https://maps.googleapis.com/maps/api/place/nearbysearch/json?location=$cen
t_lat,$cent_lng&radius=$max_distance&type=$food_types[$f]&key=AIzaSyAC9YdWF
l62YplTmM8J0OwxZZo0VXt_KpE";
        $html = file_get_contents ( $url );
        $html_dec = json_decode ( $html );
        $html_dec_vars = get_object_vars ( $html_dec );
        $results = $html_dec_vars[results];
        $status = $html_dec_vars[status];

        // print $wanted_cat;
        // print_r ( "$wanted_cat, $results, $wanted_cat" );

        // print "!!!!$wanted_cat";
        // print_r ( $results );
        // print "$wanted_cat!!!!";

        if ( $status == 'OK') {
          // $rel_monum_bef = get_metadata ( $meta_type = 'post',
$object_id =  $parents_ids_array[$n], $meta_key = 'rel_monum', $single =
true  );
          // print "religious monument before $wanted_cat:
$rel_monum_bef.";

          // $coffee_bef = get_metadata ( $meta_type = 'post', $object_id =
$parents_ids_array[$n], $meta_key = 'coffee', $single = true  );
          // print "coffee before $wanted_cat: $coffee_bef.";

          update_metadata ( $meta_type = 'post', $object_id =
$parents_ids_array[$n], $meta_key = 'coffee', $meta_value = 'at least one'
);

          // $rel_monum_aft = get_metadata ( $meta_type = 'post',
$object_id =  $parents_ids_array[$n], $meta_key = 'rel_monum', $single =
true  );
          // print "religious monument after: $rel_monum_aft.";

          // $coffee_aft = get_metadata ( $meta_type = 'post', $object_id =
```

```
$parents_ids_array[$n], $meta_key = 'coffee', $single = true  );
          // print "coffee after: $coffee_aft.";

          $f = 10000;
        } else {
          update_metadata ( $meta_type = 'post', $object_id =
$parents_ids_array[$n], $meta_key = 'coffee', $meta_value = 'none' );
        }

        $f++;
      }

      $n++;
    }
  ?>
</html>
```

4: Κατηγορία: Σημείο ενδιαφέροντος για τουρίστες

```
<!DOCTYPE html>
<html>
  <body>
    <form method="post" action="">
      <input type="text" name="maximum_distance" placeholder="in metres,
without decimals">
      <input type="submit">
    </form>
  </body>
  <?php
      // παιρνω τα id των "parents" post
      $args = array(
        'numberposts'      => -1,
        'category'         => 25,
        'orderby'          => 'date',
        'order'            => 'DESC',
        'include'          => array(),
        'exclude'          => array(),
        'meta_key'         => '',
        'meta_value'       => '',
        'post_type'        => 'post',
        'suppress_filters' => true,
        'fields'           => 'ids'
```

```
      );
      $parents_ids_array = get_posts ( $args );

      // λαμβανω τον αριθμο των track
      $parents_ids_array_size = sizeof ( $parents_ids_array );

      // για καθε track
      $n = 0;
      while ( $n <= $parents_ids_array_size - 1 ) {
        // βρισκω το id
        $current_post_id = $parents_ids_array[$n];

        // προσοχη: argument το post id, και οχι καποιο category id
        // παιρνω τα data της αντιστοιχης κατηγοριας
        $categ_data =  get_the_category( $current_post_id );

        // υπολογιζω τα category που του αντιστοιχουν
        // προσοχη: υπολογιζω των αριθμο των category
        // και οχι τον αριθμο των track
        $categ_data_size = sizeof ( $categ_data );

        // απο τα διαφορα category στα οποια μπορει να ανηκει το ΚΑΘΕ post
που ανηκει
        // στο "parents" category, παιρνω το category με format "track x"
        $i = 0;
        while ( $i <= ( $categ_data_size - 1 ) ) {
          ${"categ_data_$i"} = $categ_data[$i];
          ${"vars_categ_data_$i"} = get_object_vars ( ${"categ_data_$i"} );
          ${"categ_$i"} = ${"vars_categ_data_$i"}[name];

          if ( strpos ( $haystack = ${"categ_$i"} , $needle = 'Track' ) !==
false ) {
            $wanted_cat = ${"categ_$i"};
            $i = 10000;
          }

          $i++;
        }

        // παιρνω το id του $wanted_cat (Track x), προσοχη, οχι το id του
post, ουτε του parents category
        $wanted_cat_id = get_cat_ID ( $wanted_cat );
```

```
        // παιρνω τα id των point(τα θελω ωστε να βρω το κεντρο βαρους τους)
        $args = array(
          'numberposts'      => -1,
          'category'         => $wanted_cat_id,
          'orderby'          => 'date',
          'order'            => 'DESC',
          'include'          => array(),
          'exclude'          => array( $parents_ids_array[$n] ),
          'meta_key'         => 'codespacing_progress_map_lat',
          'meta_value'       => '',
          'post_type'        => 'post',
          'suppress_filters' => true,
          'fields'           => 'ids'
        );
        $ids_array = get_posts ( $args );

        // ουσιαστικα, λαμβανω τον αριθμο των point
        $posts_array_size = sizeof ( $ids_array );

        // για καθε point του track
        $j = 0;
        while ( $j <= $posts_array_size - 1 ) {
          // λαμβανω τις συντεταγμενες
          $post_lat = get_metadata ( $meta_type = 'post', $object_id =
$ids_array[$j], $meta_key = 'codespacing_progress_map_lat', $single = true
);
          $post_lng = get_metadata ( $meta_type = 'post', $object_id =
$ids_array[$j], $meta_key = 'codespacing_progress_map_lng', $single = true
);

          $lats_sum += $post_lat;
          $lngs_sum += $post_lng;

          $j++;
        }

        // υπολογιζω το κεντρο βαρους
        $cent_lat = $lats_sum/$posts_array_size;
        $cent_lng = $lngs_sum/$posts_array_size;

        // εκχωρω την επιθυμητη, μεγιστη (οχι μεγιστη, επιθυμητη) αποσταση που
δινει ο χρηστης
        $max_distance = $_POST['maximum_distance'];
```

```
        // λαμβανω τα point που το google βρισκει, εντος της μεγιστης,
επιθυμητης αποστασης
        $url =
"https://maps.googleapis.com/maps/api/place/nearbysearch/json?location={$ce
nt_lat},{$cent_lng}&radius={$max_distance}&type=tourist_attraction&key=AIza
SyAC9YdWFl62YplTmM8J0OwxZZo0VXt_KpE";
        $html = file_get_contents ( $url );
        $html_dec = json_decode ( $html );
        $html_dec_vars = get_object_vars ( $html_dec );
        $results = $html_dec_vars[results];

        // προσδιοριζω τον αριθμο των point που επιστρεφει το google
        $results_size = sizeof ( $results );

        // απαντω αν υπαρχει τουλαχιστον ενα point του ιχνους, το οποιο
βρισκεται κοντα
        // σε monument
        // για καθε point του google
        $m = 0;
        while ( $m <= $results_size ) {
          // λαμβανω το array με τα google types στα οποια ανηκει (το καθε
point του google)
          $result = $results[$m];
          $result_array = get_object_vars ( $result );
          $types = $result_array[types];

          // λαμβανω τον αριθμο των google types στα οποια ανηκει
          $types_size = sizeof ( $types );

          // για καθε google type που ελαβα
          $k = 0;
          while ( $k <= $types_size - 1 ) {
            // συγκρινω με το αναζητουμενο type, και αν βρω ταυτοτητα
            if ( $types[$k] == 'tourist_attraction') {
              // βαζω στο field την τιμη 1
              update_metadata ( $meta_type = 'post', $object_id =
$parents_ids_array[$n], $meta_key = 'monum', $meta_value = 'at least one'
);
              $k = 10000;
              $m = 10000;
            }
            $k++;
```

```
          }

          // αν δεν βρω tourist attraction, βαζω 0
          if ( $k < 10000 ) {
            update_metadata ( $meta_type = 'post', $object_id =
$parents_ids_array[$n], $meta_key = 'monum', $meta_value = 'none' );
          }

          $m++;
        }

        $n++;
      }
    ?>
</html>
```

5: Κατηγορία: Θρησκευτικό κτίριο

```
<!DOCTYPE html>
<html>
  <body>
    <form method="post" action="">
      <input type="text" name="maximum_distance" placeholder="in metres,
without decimals">
      <input type="submit">
    </form>
  </body>
  <?php
    // εκχωρω την επιθυμητη, μεγιστη (οχι μεγιστη, επιθυμητη) αποσταση που δινει
ο χρηστης
    $max_distance = $_POST['maximum_distance'];

    // παιρνω τα id των "parents" post
    $args = array(
      'numberposts'      => -1,
      'category'         => 25,
      'orderby'          => 'date',
      'order'            => 'DESC',
      'include'          => array(),
      'exclude'          => array(),
      'meta_key'         => '',
      'meta_value'       => '',
```

```
      'post_type'        => 'post',
      'suppress_filters' => true,
      'fields'           => 'ids'
    );
    $parents_ids_array = get_posts ( $args );

    // λαμβανω τον αριθμο των track
    $parents_ids_array_size = sizeof ( $parents_ids_array );

    // φτιαχνω array με τα food types
    $rel_monum_types = array ( "church", "hindu_temple", "mosque",
"synagogue" );

    // print $rel_monum_types[0];
    // print $rel_monum_types[1];

    $rel_monum_size = sizeof ( $rel_monum_types );

    // για καθε track
    $n = 0;
    while ( $n <= $parents_ids_array_size - 1 ) {

      // βρισκω το id
      $current_post_id = $parents_ids_array[$n];

      // προσοχη: argument το post id, και οχι καποιο category id
      // παιρνω τα data της αντιστοιχης κατηγοριας
      $categ_data =  get_the_category( $current_post_id );

      // υπολογιζω τα category που του αντιστοιχουν
      // προσοχη: υπολογιζω των αριθμο των category
      // και οχι τον αριθμο των track
      $categ_data_size = sizeof ( $categ_data );

      // απο τα διαφορα category στα οποια μπορει να ανηκει το ΚΑΘΕ post που
ανηκει
      // στο "parents" category, παιρνω το category με format "track x"
      $i = 0;
      while ( $i <= ( $categ_data_size - 1 ) ) {
        ${"categ_data_$i"} = $categ_data[$i];
        ${"vars_categ_data_$i"} = get_object_vars ( ${"categ_data_$i"} );
        ${"categ_$i"} = ${"vars_categ_data_$i"}[name];
```

```
        if ( strpos ( $haystack = ${"categ_$i"} , $needle = 'Track' ) !==
false ) {
          $wanted_cat = ${"categ_$i"};
          $i = 10000;
        }

        $i++;
      }

      // παιρνω το id του $wanted_cat (Track x), προσοχη, οχι το id του post,
ουτε του parents category
      $wanted_cat_id = get_cat_ID ( $wanted_cat );

      // παιρνω τα id των point(τα θελω ωστε να βρω το κεντρο βαρους τους)
      $args = array(
        'numberposts'      => -1,
        'category'         => $wanted_cat_id,
        'orderby'          => 'date',
        'order'            => 'DESC',
        'include'          => array(),
        'exclude'          => array( $parents_ids_array[$n] ),
        'meta_key'         => 'codespacing_progress_map_lat',
        'meta_value'       => '',
        'post_type'        => 'post',
        'suppress_filters' => true,
        'fields'           => 'ids'
      );
      $ids_array = get_posts ( $args );

      // ουσιαστικα, λαμβανω τον αριθμο των point
      $posts_array_size = sizeof ( $ids_array );

      // για καθε point του track
      $j = 0;
      $lats_sum = 0;
      $lngs_sum = 0;
      while ( $j <= $posts_array_size - 1 ) {
        // λαμβανω τις συντεταγμενες
        $post_lat = get_metadata ( $meta_type = 'post', $object_id =
$ids_array[$j], $meta_key = 'codespacing_progress_map_lat', $single = true
);
        $post_lng = get_metadata ( $meta_type = 'post', $object_id =
$ids_array[$j], $meta_key = 'codespacing_progress_map_lng', $single = true
```

```
);

        $lats_sum += $post_lat;
        $lngs_sum += $post_lng;

        $j++;
      }

      // υπολογιζω το κεντρο βαρους
      $cent_lat = $lats_sum/$posts_array_size;
      $cent_lng = $lngs_sum/$posts_array_size;

      // για καθε type
      $f = 0;
      while ( $f <= $rel_monum_size - 1 ) {
        // print $cent_lat;
        // print $cent_lng;

        // print $max_distance;

        // λαμβανω τα point που το google βρισκει, εντος της μεγιστης,
επιθυμητης αποστασης
        $url =
"https://maps.googleapis.com/maps/api/place/nearbysearch/json?location=$cen
t_lat,$cent_lng&radius=$max_distance&type=$rel_monum_types[$f]&key=AIzaSyAC
9YdWFl62YplTmM8J0OwxZZo0VXt_KpE";
        // print $url;
        // print "rn";
        // print $rel_monum_types[$f];

        $html = file_get_contents ( $url );
        $html_dec = json_decode ( $html );
        $html_dec_vars = get_object_vars ( $html_dec );

        // $results = $html_dec_vars[results];
        $status = $html_dec_vars[status];

        // print $wanted_cat;
        // print $rel_monum_types[$f];
        // print "$cent_lat  ";
        // print $cent_lng;
        // print $status;
```

```
        // print $wanted_cat;

        if ( $status == 'OK') {
          // $cof_bef = get_metadata ( $meta_type = 'post', $object_id =
$parents_ids_array[$n], $meta_key = 'coffee', $single = true  );
          // print "coffee before, {$wanted_cat}: {$cof_bef}.";

          // $rel_bef = get_metadata ( $meta_type = 'post', $object_id =
$parents_ids_array[$n], $meta_key = 'rel_monum', $single = true );
          // print "rel before, {$wanted_cat}: {$rel_bef}.";

          update_metadata ( $meta_type = 'post', $object_id =
$parents_ids_array[$n], $meta_key = 'rel_monum', $meta_value = 'at least
one' );

          // $cof_aft = get_metadata ( $meta_type = 'post', $object_id =
$parents_ids_array[$n], $meta_key = 'coffee', $single = true  );
          // print "coffee after: {$cof_aft}.";

          // $rel_aft = get_metadata ( $meta_type = 'post', $object_id =
$parents_ids_array[$n], $meta_key = 'rel_monum', $single = true );
          // print "rel after, {$wanted_cat}: {$rel_aft}.";

          $f = 10000;
        } else {
          update_metadata ( $meta_type = 'post', $object_id =
$parents_ids_array[$n], $meta_key = 'rel_monum', $meta_value = 'none' );
        }

        $f++;
      }

      $n++;
    }
  ?>
</html>
```

Παράρτημα Γ : Υπολογισμός μέσης κλίσης

```
 <!DOCTYPE html>
<html>
  <body>
    <?php
      /**
         * Calculates the great-circle distance between two points, with
         * the Vincenty formula.
         * @param float $latitudeFrom Latitude of start point in [deg
decimal]
         * @param float $longitudeFrom Longitude of start point in [deg
decimal]
         * @param float $latitudeTo Latitude of target point in [deg
decimal]
         * @param float $longitudeTo Longitude of target point in [deg
decimal]
         * @param float $earthRadius Mean earth radius in [m]
         * @return float Distance between points in [m] (same as
earthRadius)
         */

        function vincentyGreatCircleDistance($latitudeFrom, $longitudeFrom,
$latitudeTo, $longitudeTo, $earthRadius = 6371000)
        {
          // convert from degrees to radians
          $latFrom = deg2rad($latitudeFrom);
          $lonFrom = deg2rad($longitudeFrom);
          $latTo = deg2rad($latitudeTo);
          $lonTo = deg2rad($longitudeTo);

          $lonDelta = $lonTo - $lonFrom;
          $a = pow(cos($latTo) * sin($lonDelta), 2) + pow(cos($latFrom) *
sin($latTo) - sin($latFrom) * cos($latTo) * cos($lonDelta), 2);
          $b = sin($latFrom) * sin($latTo) + cos($latFrom) * cos($latTo) *
cos($lonDelta);

          $angle = atan2(sqrt($a), $b);
          return $angle * $earthRadius;
        }

      // παιρνω τα id των posts που ανηκουν στο parents
```

```
      $parents_categ = 25;
      $parents_args = array(
        'numberposts'      => -1,
        'category'         => $parents_categ,
        'orderby'          => 'date',
        'order'            => 'DESC',
        'include'          => array(),
        'exclude'          => array(),
        'meta_key'         => 'codespacing_progress_map_lat',
        'meta_value'       => '',
        'post_type'        => 'post',
        'suppress_filters' => true,
        'fields'           => 'ids'
      );
      $parents_ids_array = get_posts ( $parents_args );

      $parents_posts_array_size = sizeof ( $parents_ids_array );

      // για καθε parents point
      $j = 0;
      while ( $j <= $parents_posts_array_size - 1 ) {

        // βρισκω το id
        $current_post_id = $parents_ids_array[$j];

        // παινρω τα category data
        $categ_data =  get_the_category( $current_post_id );

        // υπολογιζω τα category που του αντιστοιχουν
        $categ_data_size = sizeof ( $categ_data );

        // επιλεγω το category με ονομα τυπου Track x
        $k = 0;
        while ( $k <= ( $categ_data_size - 1 ) ) {
          ${"categ_data_$k"} = $categ_data[$k];
          ${"vars_categ_data_$k"} = get_object_vars ( ${"categ_data_$k"} );
          ${"categ_$k"} = ${"vars_categ_data_$k"}[name];

          if ( strpos ( $haystack = ${"categ_$k"} , $needle = 'Track' ) !==
false ) {
            $wanted_cat = ${"categ_$k"};
            // print ${"categ_$i"};
            $k = 10000;
```

```
          }

          $k++;
        }

        // παιρνω το id του $wanted_cat (Track x), προσοχη, οχι το id του
post, ουτε του parents category
        $wanted_cat_id = get_cat_ID ( $wanted_cat );

        // για το category τυπου Track x βρισκω τα id των points που ανηκουν
στο category, εκτος του $j parent point
        $args = array(
          'numberposts'      => -1,
          'category'         => $wanted_cat_id,
          'orderby'          => 'date',
          'order'            => 'DESC',
          'include'          => array(),
          'exclude'          => array($parents_ids_array[$j]),
          'meta_key'         => 'codespacing_progress_map_lat',
          'meta_value'       => '',
          'post_type'        => 'post',
          'suppress_filters' => true,
          'fields'           => 'ids'
        );
        $ids_array = get_posts ( $args );

        // υπολογιζω τον αριθμο των point που αντιστοιχουν στο ιχνος Track x
        $posts_array_size = sizeof ( $ids_array );

        // για καθε point του ιχνους Track x
        $u = 0;
        $slope_sum = 0;
        while ( $u <= $posts_array_size - 1 ) {
          // λαμβανω τις συντεταγμενες του u point
          $post_lat = get_metadata ( $meta_type = 'post', $object_id =
$ids_array[$u], $meta_key = 'codespacing_progress_map_lat', $single = true
);
          $post_lng = get_metadata ( $meta_type = 'post', $object_id =
$ids_array[$u], $meta_key = 'codespacing_progress_map_lng', $single = true
);

          // λαμβανω τις συντεταγμενες του u+1 point
          $post_lat_next = get_metadata ( $meta_type = 'post', $object_id =
```

```
$ids_array[$u+1], $meta_key = 'codespacing_progress_map_lat', $single =
true  );
          $post_lng_next = get_metadata ( $meta_type = 'post', $object_id =
$ids_array[$u+1], $meta_key = 'codespacing_progress_map_lng', $single =
true  );

          // λαμβανω το elevation του u point
          $url =
"https://maps.googleapis.com/maps/api/elevation/json?locations={$post_lat},
{$post_lng}&key=AIzaSyAC9YdWFl62YplTmM8J0OwxZZo0VXt_KpE";
          $html = file_get_contents ( $url );
          $html_dec = json_decode ( $html );
          $html_dec_vars = get_object_vars ( $html_dec );
          $results = $html_dec_vars[results];
          $results_0 = $results[0];
          $results_0_vars = get_object_vars ( $results_0 );
          $elevation = $results_0_vars[elevation];

          // λαμβανω το elevation του u+1 point
          $url =
"https://maps.googleapis.com/maps/api/elevation/json?locations={$post_lat_n
ext},{$post_lng_next}&key=AIzaSyAC9YdWFl62YplTmM8J0OwxZZo0VXt_KpE";
          $html = file_get_contents ( $url );
          $html_dec = json_decode ( $html );
          $html_dec_vars = get_object_vars ( $html_dec );
          $results = $html_dec_vars[results];
          $results_0 = $results[0];
          $results_0_vars = get_object_vars ( $results_0 );
          $elevation_next = $results_0_vars[elevation];

          // λαμβανω την αποσταση των point u, και u+1
          $distance = vincentyGreatCircleDistance ( $post_lat, $post_lng,
$post_lat_next, $post_lng_next, $earthRadius = 6371000 );

          // υπολογιζω την κλιση του τμηματος u, u+1
          $slope = abs ( $elevation_next - $elevation )/$distance;

          $slope_sum += $slope;

          $u++;
        }

        // βρισκω το mean_slope του Track x
```

```
        $mean_slope = 100 * $slope_sum/$posts_array_size;

        $mean_slope = round ( $mean_slope );

        // print $mean_slope;

        // βρισκω το mean_slope
        // $mean_slope_before = get_metadata( $meta_type = 'post',
$object_id = $current_post_id, $meta_key = 'slope', $single = true );
        // print $mean_slope_before;

        if ( $mean_slope != 0 ) {
          update_metadata ( $meta_type = 'post', $object_id =
$current_post_id, $meta_key = 'slope', $meta_value = $mean_slope );
        } else {
          update_metadata ( $meta_type = 'post', $object_id =
$current_post_id, $meta_key = 'slope', $meta_value = 1 );
        }

        // $mean_slope_after = get_metadata( $meta_type = 'post',
$object_id = $current_post_id, $meta_key = 'slope', $single = true );
        // print $mean_slope_after;

        $j++;
      }
    ?>
  </body>
</html>
```

Παράρτημα Δ : Ανανέωση θερμοκρασίας

```
<!DOCTYPE html>
<html>
  <body>

  <?php
    $args = array(
      'numberposts'      => -1,
      'category'         => 25,
      'orderby'          => 'date',
      'order'            => 'DESC',
      'include'          => array(),
      'exclude'          => array(),
      'meta_key'         => '',
      'meta_value'       => '',
      'post_type'        => 'post',
      'suppress_filters' => true,
      'fields'           => 'ids'
    );
    // παιρνω τα id των ιχνών
    $get_posts_array = get_posts ( $args );

    // μετραω τον αριθμο των ιχνων
    $get_posts_array_size = sizeof ( $get_posts_array );

    // για καθε ιχνος
    $i = 0;
    while ( $i <= $get_posts_array_size - 1 ) {
      // λαμβανω τις συντεταγμενες του σημειου που συμβολιζει το ιχνος
      $post_lat = get_metadata ( $meta_type = 'post', $object_id =
$get_posts_array[$i], $meta_key = 'codespacing_progress_map_lat', $single =
true  );
      $post_lng = get_metadata ( $meta_type = 'post', $object_id =
$get_posts_array[$i], $meta_key = 'codespacing_progress_map_lng', $single =
true  );

      // λαμβάνω την θερμοκρασια του σημειου
      $url =
"https://api.openweathermap.org/data/2.5/onecall?lat={$post_lat}&lon={$post_lng
}&%20exclude=minutely,hourly,daily&appid=71c1d8472d62b82e44cc8a4fbab34f9a&units
=metric";
      $html = file_get_contents ( $url );
      $html_dec = json_decode ( $html );
```

```
      $html_dec_vars = get_object_vars ( $html_dec );
      $current = $html_dec_vars[current];
      $current_vars = get_object_vars ( $current );
      $temp = $current_vars[temp];
      $temp = round ( $temp );
      // $temp = floor ( $temp );
      // print ( gettype ( $temp ) );

      // $temp_before = get_metadata( $meta_type = 'post', $object_id =
$get_posts_array[$i], $meta_key = 'temp', $single = true );
      // print "{$temp_before}||";

      // εκχωρω την θερμοκρασια
      update_metadata ( $meta_type = 'post', $object_id = $get_posts_array[$i],
$meta_key = 'temp', $meta_value = $temp );

      // $temp_now = get_metadata( $meta_type = 'post', $object_id =
$get_posts_array[$i], $meta_key = 'temp', $single = true );
      // print "{$temp_now}||";

      $i++;
    }

    $args_2 = array(
      'numberposts'      => -1,
      'category'         => 0,
      'category__not_in' => 25,
      'orderby'          => 'date',
      'order'            => 'DESC',
      'include'          => array(),
      'exclude'          => array(),
      'meta_key'         => '',
      'meta_value'       => '',
      'post_type'        => 'post',
      'suppress_filters' => true,
      'fields'           => 'ids'
    );
    $child_ids = get_posts ( $args_2 );

    $child_ids_number = sizeof ( $child_ids );

    // $j = 0;
    // while ( $j <= $child_ids_number - 1 ) {
    //   delete_metadata ( $meta_type = 'post', $object_id = $child_ids[$j],
$meta_key = 'temp' );
```

```
    //    $j++;
    // }
  ?>

  </body>
</html>
```

Παράρτημα Ε : Προβολή μηκοτομής ίχνους

```
<!DOCTYPE html>
<html>
  <head>
    <title>Showing Elevation Along a Path</title>
    <!-- n: τα scripts απο το elevation. -->
    <script
src="https://polyfill.io/v3/polyfill.min.js?features=default"></script>
    <script src="https://www.google.com/jsapi"></script>
    <script

src="https://maps.googleapis.com/maps/api/js?key=AIzaSyAC9YdWFl62YplTmM8J0OwxZZ
o0VXt_KpE&callback=initMap&libraries=&v=weekly"
      defer
    ></script>

    <!-- n:  -->
    <!-- <script src="http://maps.google.com/maps/api/js?sensor=false"
          type="text/javascript"></script> -->
    <style type="text/css">
      /* Always set the map height explicitly to define the size of the div
       * element that contains the map. */
      #map {
        height: 100%;
        /* height: 300px; */
      }

      /* Optional: Makes the sample page fill the window. */
      html,
      body {
        height: 100%;
        /* height: 300px; */
        margin: 0;
        padding: 0;
      }

      #legend {
        font-family: Arial, sans-serif;
        background: #fff;
        padding: 10px;
        margin: 10px;
        border: 3px solid #000;
      }
```

```
    </style>
    <script>
      // elev-along
        "use strict";

        // Load the Visualization API and the columnchart package.
        google.load("visualization", "1", {
          packages: ["columnchart"]
        });

      <?php
        // elev-along
          // n: αξιοποιηση στην ληψη του category της μορφης track x
          // get wp-term object
          $categ_data =  get_the_category();
          // $categ_wp_obj = $categ[0];

          // n: αξιοποιηση στην ληψη του category της μορφης track x
          $categ_data_size = sizeof ( $categ_data );

          // n: ωστε να παρουμε το id του category
            // απο τα διαφορα category στα οποια μπορει να ανηκει το present
post, παιρνω το category με format "track x"
              $i = 0;
              while ( $i <= ( $categ_data_size - 1 ) ) {
                ${"categ_data_$i"} = $categ_data[$i];
                ${"vars_categ_data_$i"} = get_object_vars ( ${"categ_data_$i"}
);
                ${"categ_$i"} = ${"vars_categ_data_$i"}[name];

                if ( strpos ( $haystack = ${"categ_$i"} , $needle = 'Track' )
!== false ) {
                  $wanted_cat = ${"categ_$i"};

                  $i = 10000;
                }

                $i++;
              }

          // n: ωστε να καθορισουμε το args
          // παιρνω το id του category
          $wanted_cat_id = get_cat_ID( $wanted_cat );

          // n: ωστε να παρουμε τα id των σημειων του τρεχοντος ιχνους
```

```
          $args = array(
            'numberposts'      => -1,
            'category'         => $wanted_cat_id,
            'orderby'          => 'date',
            'order'            => 'DESC',
            'include'          => array(),
            'exclude'          => get_the_ID(),
            'meta_key'         => '',
            'meta_value'       => '',
            'post_type'        => 'post',
            'suppress_filters' => true,
            'fields'           => 'ids'
          );
          // n: ωστε να παρουμε τις συντεταγμενες των σημειων του τρεχοντος ιχνους
          $ids_array = get_posts ( $args );

          // print_r ( $ids_array );

          // n: ωστε να παρουμε τις συντεταγμενες των σημειων του τρεχοντος ιχνους
          $posts_array_size = sizeof ( $ids_array );

          // βαζουμε σε 2 πινακες, τα μηκη και τα πλατη των σημειων του τρεχοντος
ιχνους.
            for ($i = 0; $i < $posts_array_size; ++$i) {
              $lats[] = get_metadata ( $meta_type = 'post', $object_id =
$ids_array[$i], $meta_key = 'codespacing_progress_map_lat', $single = true );
              $lngs[] = get_metadata ( $meta_type = 'post', $object_id =
$ids_array[$i], $meta_key = 'codespacing_progress_map_lng', $single = true );
            }

          $lats_json = json_encode ( $lats );
          $lngs_json = json_encode ( $lngs );
        // show-pois
          // για καθε point του track
          $j = 0;
          $lats_sum = 0;
          $lngs_sum = 0;
          while ( $j <= $posts_array_size - 1 ) {
            // λαμβανω τις συντεταγμενες
            $post_lat = get_metadata ( $meta_type = 'post', $object_id =
$ids_array[$j], $meta_key = 'codespacing_progress_map_lat', $single = true  );
            $post_lng = get_metadata ( $meta_type = 'post', $object_id =
$ids_array[$j], $meta_key = 'codespacing_progress_map_lng', $single = true  );

            $lats_sum += $post_lat;
```

```
            $lngs_sum += $post_lng;

            $j++;
          }

          // υπολογιζω το κεντρο βαρους
          $cent_lat = $lats_sum/$posts_array_size;
          $cent_lng = $lngs_sum/$posts_array_size;


          $food_types = array ( 0 => "bakery", 1 => "meal_takeaway", 2 =>
"restaurant" );
          $police = array ( 0 => "police" );
          $coffee_types = array ( 0 => "bar", 1 => "cafe" );
          $monuments = array ( 0 => "tourist_attraction" );
          $rel_monum_types = array ( 0 =>"church", 1 => "hindu_temple", 2 =>
"mosque", 3 => "synagogue" );

          $type_set_arr = array (
            0 => $food_types,
            1 => $police,
            2 => $coffee_types,
            3 => $monuments,
            4 => $rel_monum_types
          );

          // μεταφερε σε μεταβλητη, την αποσταση food που ο χρηστης εβαλε
          // στην φορμα
          $food_dist = $_POST['food_maximum_distance'];
          $pol_dist = $_POST['police_maximum_distance'];
          $coffee_dist = $_POST['coffee_maximum_distance'];
          $monum_dist = $_POST['monument_maximum_distance'];
          $rel_mon_dist = $_POST['religious_monument_maximum_distance'];

          $dist_crit_arr = array (
            0 => $food_dist,
            1 => $pol_dist,
            2 => $coffee_dist,
            3 => $monum_dist,
            4 => $rel_mon_dist
          );

          // $c = 0;
          // print $k;
```

```
          // για καθε sup type
          $t = 0;
          while ( $t < 5 ) {
            // για καθε type
            $k = 0;
            while ( $k < sizeof ( $type_set_arr[$t] ) ) {
              // print $k;
              // παρε τα point που το google βρισκει, εντος της μεγιστης,
επιθυμητης αποστασης.
              // εκχωρησε το url σε μεταβλητη
              $url =
"https://maps.googleapis.com/maps/api/place/nearbysearch/json?location=$cent_la
t,$cent_lng&radius=$dist_crit_arr[$t]&type={$type_set_arr[$t][$k]}&key=AIzaSyAC
9YdWFl62YplTmM8J0OwxZZo0VXt_KpE";
              // print $url;
              // print $cent_lat;
              // print $cent_lng;
              // print $food_dist;
              // print $food_types[$k];

              // print_r ( $type_set_arr[$t][$k] );

              // βαλε το file σε string
              $html = file_get_contents ( $url );
              // print $html;

              // αποκωδικοποιησε το json string (μαλλον σε object)
              $html_dec = json_decode ( $html );

              // παρε τα properties του object
              $html_dec_vars = get_object_vars ( $html_dec );

              // παρε το status
              $status = $html_dec_vars[status];
              // print $status;

              // αν το google εχει επιστρεψει τουλαχιστον ενα σημειο
              if ( $status == "OK" ) {
                // παρε το results
                $results = $html_dec_vars[results];

                // print_r ( $results );

                // μετρα το μεγεθος του results
                $res_size = sizeof ( $results );
```

```
                // print $res_size;

                $l = 0;
                // για καθε σημειο του results
                while ( $l < $res_size ) {
                  // παρε τα results, απο το google response
                  // ${"res_$l"} = $results[$l];
                  // ${"res_$l"} = $results;
                  // print_r ( ${"res_$l"} );
                  $res = $results[$l];

                  // κανε το array
                  // ${"res_arr_$l"} = get_object_vars ( ${"res_$l"} );
                  $res_arr = get_object_vars ( $res );
                  // print_r ( ${"res_arr_$l"} );

                  // παρε το geometry object του array
                  // ${"geom_res_arr_$l"} = ${"res_arr_$l"}[geometry];
                  $geom_res_arr = $res_arr[geometry];
                  // print_r ( ${"geom_res_arr_$l"} );

                  // μετατρεψε το geometry σε array
                  // ${"geom_arr_$l"} = get_object_vars ( ${"geom_res_arr_$l"}
);
                  $geom_arr = get_object_vars ( $geom_res_arr );
                  // print_r ( ${"geom_arr_$l"} );

                  // παρε το location object (συντεταγμενες σημειου), απο το
geometry array
                  // ${"loc_$l"} = ${"geom_arr_$l"}[location];
                  $loc = $geom_arr[location];

                  // μετάτρεψέ το σε array
                  // ${"loc_arr_$l"} = get_object_vars ( ${"loc_$l"} );
                  $loc_arr = get_object_vars ( $loc );
                  // print_r ( ${"loc_arr_$l"} );

                  // $poi_lats = ${"loc_arr_$l"}

                  // $pois_locs[$k] = json_encode ( ${"loc_arr_$l"} );

                  $pois_lats[] = $loc_arr[lat];
                  $pois_lngs[] = $loc_arr[lng];

                  // καταγραψε το type του σημειου
```

```
                $type[] = $type_set_arr[$t][$k];

                $check_food = in_array ( $needle = $type_set_arr[$t][$k],
$haystack = $food_types );
                // $check_police = $in_array ( $needle =
$type_set_arr[$t][$k], $haystack = $police );

                $check_police = in_array ( $needle = $type_set_arr[$t][$k],
$haystack = $police );

                $check_coffee = in_array ( $needle = $type_set_arr[$t][$k],
$haystack = $coffee_types );

                $check_monum = in_array ( $needle = $type_set_arr[$t][$k],
$haystack = $monuments );

                if ( $check_food == 1 ) {
                  $s_type[] = "food";
                } elseif ( $check_police == 1 ) {
                  $s_type[] = "police";
                } elseif ( $check_coffee == 1 ) {
                  $s_type[] = "coffee";
                } elseif ( $check_monum == 1 ) {
                  $s_type[] = "monument";
                } else {
                  $s_type[] = "religious_monument";
                }

                $l++;
                // $c++;
              }

              // $c += $l;
              // $pois_lats[$k] = ${"loc_arr_$l"}[0];
              // $pois_lngs[$k] = ${"loc_arr_$l"}[0];
            }

            $k++;
          }

          $t++;
        }

        // print_r ( $s_type );
```

```
            $p_lats_j = json_encode ( $pois_lats );
            $p_lngs_j = json_encode ( $pois_lngs );

            $type_j = json_encode ( $type );
            $s_type_j = json_encode ( $s_type );

            // $p_lats_bak = json_encode ( $pois_lats_1 );
            // $p_lngs_bak = json_encode ( $pois_lngs_1 );
        ?>

        // συνεχεια JS
          // elev-along
            //
https://stackoverflow.com/questions/2383484/how-to-create-a-dynamic-object-in-a
-loop
            // tomalak

            var lats_json = <?php echo $lats_json; ?>;
            // document.write(lats_json);

            var size = lats_json.length;

            var lngs_json = <?php echo $lngs_json; ?>;
            // document.write(Array.isArray(lngs_json));
            // document.write(lngs_json[0]);

            var path = [];
            for (var i = 0; i < size; i++) {
              path.push({
                lat: Number(lats_json[i]),
                lng: Number(lngs_json[i])
              });
            }

            // var locs$json = <php echo $locs_json; ?>;
            // document.write(locs$json);

            var p$lats$j = <?php echo $p_lats_j; ?>;
            var p$lngs$j = <?php echo $p_lngs_j; ?>;

            // console.log(p$lats$j);

            var type_j = <?php echo $type_j; ?>;
            var s$type$j = <?php echo $s_type_j; ?>;
```

```
  // console.log(type_j);
  // console.log(s$type$j);

  // document.write(2);

  // console.log(pois);

// ορισε την initMap.
// η initMap καλει την displayPathElevation, της οποιας ο ορισμος
// υπαρχει παρακατω
function initMap() {
  // document.write(2);

  // console.log(pois);
  // document.write(2);

  const map = new google.maps.Map(document.getElementById("map"), {
    zoom: 8,
    center: path[1],
    mapTypeId: "terrain"
  }); // Create an ElevationService.

  // show-pois
    const icons = {
      food: {
        name: "food",
        icon: "http://maps.google.com/mapfiles/kml/pal2/icon38.png",
      },
      coffee: {
        name: "cofee",
        icon: "http://maps.google.com/mapfiles/kml/pal2/icon54.png",
      },
      police: {
        name: "police",
        icon: "http://maps.google.com/mapfiles/kml/pal2/icon24.png",
      },
      monument: {
        name: "tourist attraction",
        icon: "http://maps.google.com/mapfiles/kml/pal2/icon2.png",
      },
      religious_monument: {
        name: "religious monument",
        icon: "http://maps.google.com/mapfiles/kml/pal2/icon3.png"
      }
    };
```

```
            //
map.controls[google.maps.ControlPosition.LEFT_BOTTOM].push(legend);

            // φτιαξε το features
            var i;
            const features = [];
            for (i = 0; i < p$lats$j.length; i++) {
              features.push({
                position: {
                  lat: Number(p$lats$j[i]),
                  lng: Number(p$lngs$j[i]),
                },
                type: s$type$j[i]
              });
              console.log(s$type$j[i]);
              // document.write(s$type$j[i]);
            }

            // const features = [];

            // // δημιουργησε το features
            // p$lats$j.forEach(features.push(
            //   position:
            // ))

            // console.log(features);

            // console.log(features[1]);

            // feature$1 = features[1];

            // const {feature$1} = {}

            // icon$1 = icons{1};

            // icon$1 = icons[1];
            // var icon$1 = icons["food"];

            // console.log(icon$1);

            // φτιαξε ενα marker, για καθε στοιχειο του features.
            features.forEach((feature) => {
              new google.maps.Marker({
                position: feature.position,
```

```
                // n: ισως πρεπει το feature, να αντιστοιχει στο ονομα εκτος των επι
μερους
                // objects του icons
                icon: icons[feature.type].icon,
                map: map,
              });
              // console.log(feature);
            });

            const legend = document.getElementById("legend");

            // n: νμζω καθοριζει το περιεχομενο του υπομνηματος.
            // δλδη, μπορει να περιεχει λαθος, αλλα τα σημεια να υπαρχουν στον
χαρτη.
            for (const key in icons) {
              const type = icons[key];
              const name = type.name;
              const icon = type.icon;
              // q: ποιό ειναι το παρακατω div?
              const div = document.createElement("div");
              div.innerHTML = '<img src="' + icon + '"> ' + name;
              legend.appendChild(div);
            }

            //
map.controls[google.maps.ControlPosition.LEFT_BOTTOM].push(legend);

            const elevator = new google.maps.ElevationService(); // Draw the
path, using the Visualization API and the Elevation service.

            displayPathElevation(path, elevator, map);
          }

        // elev-along
          // μοιαζει να καλει την plotElevation
          function displayPathElevation(path, elevator, map) {
            // Display a polyline of the elevation path.
            new google.maps.Polyline({
              path: path,
              strokeColor: "#0000CC",
              strokeOpacity: 0.4,
              map: map
            }); // Create a PathElevationRequest object using this array.
            // Ask for 256 samples along that path.
            // Initiate the path request.
```

```
            elevator.getElevationAlongPath(
              {
                path: path,
                samples: 2
              },
              plotElevation
            );
          } // Takes an array of ElevationResult objects, draws the path on the
map
          // and plots the elevation profile on a Visualization API
ColumnChart.

          function plotElevation(elevations, status) {
            const chartDiv = document.getElementById("elevation_chart");

            if (status !== "OK") {
              // Show the error code inside the chartDiv.
              chartDiv.innerHTML =
                "Cannot show elevation: request failed because " + status;
              return;
            } // Create a new chart in the elevation_chart DIV.

            const chart = new google.visualization.ColumnChart(chartDiv); //
Extract the data from which to populate the chart.
            // Because the samples are equidistant, the 'Sample'
            // column here does double duty as distance along the
            // X axis.

            const data = new google.visualization.DataTable();
            data.addColumn("string", "Sample");
            data.addColumn("number", "Elevation");

            for (let i = 0; i < elevations.length; i++) {
              data.addRow(["", elevations[i].elevation]);
            } // Draw the chart using the data within its DIV.

            chart.draw(data, {
              height: 150,
              legend: "none",
              titleY: "Elevation (m)"
            });
          }
    </script>
  </head>
```

```
  <body>
    <form method="post" action="">
      <input type="text" name="food_maximum_distance" placeholder="food: in
metres, without decimals">
      <input type="text" name="police_maximum_distance" placeholder="police: in
metres, without decimals">
      <input type="text" name="coffee_maximum_distance" placeholder="coffee: in
metres, without decimals">
      <input type="text" name="monument_maximum_distance" placeholder="tourist
attraction: in metres, without decimals">
      <input type="text" name="religious_monument_maximum_distance"
placeholder="religious monument: in metres, without decimals">
      <input type="submit">
    </form>
    <div>
      <div id="map" style="height:250px;"></div>
      <div id="legend"><h3>Legend</h3></div>
    </div>
    <div id="elevation_chart"></div>
  </body>
</html>
```