ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

Σχολή Τεχνολογίας

Τμήμα Ψηφιακών Συστημάτων

Ανάπτυξη εφαρμογής για την συλλογή , επεξεργασία και οπτικοποίηση μετεωρολογικών δεδομένων

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Ανδρέας Γιαννακόπουλος (ΑΜ: 3120071)

Επιβλέπων: ***Ονούφριος Χαραλάμπους, Αναπληρωτής Καθηγητής***

ΛΑΡΙΣΑ, Οκτώβριος 2024

UNIVERSITY OF THESSALY

School of Technology

Digital Systems Department

Development of applications for collection, processing and visualization of open meteorological forecasts

BACHELOR THESIS

Andreas Giannakopoulos (RN: 3120071)

Supervisor: ***Onoufrios Haralampous, Associate Professor***

LARISSA, October 2024

Υπεύθυνη Δήλωση περί Ακαδημαϊκής Δεοντολογίας και Πνευματικών Δικαιωμάτων

Με πλήρη επίγνωση των συνεπειών του νόμου περί πνευματικών δικαιωμάτων, δηλώνω ρητά ότι η παρούσα πτυχιακή εργασία, καθώς και τα ηλεκτρονικά αρχεία και πηγαίοι κώδικες που αναπτύχθηκαν ή τροποποιήθηκαν στα πλαίσια αυτής της εργασίας, αποτελεί αποκλειστικά προϊόν προσωπικής μου εργασίας, δεν προσβάλλει κάθε μορφής δικαιώματα διανοητικής ιδιοκτησίας, προσωπικότητας και προσωπικών δεδομένων τρίτων, δεν περιέχει έργα/εισφορές τρίτων για τα οποία απαιτείται άδεια των δημιουργών/δικαιούχων και δεν είναι προϊόν μερικής ή ολικής αντιγραφής, οι πηγές δε που χρησιμοποιήθηκαν περιορίζονται στις βιβλιογραφικές αναφορές και μόνον, και πληρούν τους κανόνες της επιστημονικής παράθεσης. Τα σημεία όπου έχω χρησιμοποιήσει ιδέες, κείμενο, αρχεία ή/και πηγές άλλων συγγραφέων, αναφέρονται ευδιάκριτα στο κείμενο με την κατάλληλη παραπομπή και η σχετική αναφορά περιλαμβάνεται στο τμήμα των βιβλιογραφικών αναφορών με πλήρη περιγραφή.

Αναλαμβάνω πλήρως, ατομικά και προσωπικά, όλες τις νομικές και διοικητικές συνέπειες που δύναται να προκύψουν στην περίπτωση κατά την οποία αποδειχθεί, διαχρονικά, ότι η εργασία αυτή ή τμήμα της δεν μου ανήκει διότι είναι προϊόν λογοκλοπής.

Ο Δηλών

(Υπογραφή)

Ανδρέας Γιαννακόπουλος

Ημερομηνία

03/10/2024

Εγκρίνεται από την Επιτροπή Εξέτασης:

Επιβλέπων	**Ονούφριος Χαραλάμπους** Αναπληρωτής Καθηγητής Τμήμα Ψηφιακών Συστημάτων, Πανεπιστήμιο Θεσσαλίας
Μέλος	**Κωνσταντίνος Χαϊκάλης** Αναπληρωτής Καθηγητής Τμήμα Ψηφιακών Συστημάτων, Πανεπιστήμιο Θεσσαλίας
Μέλος	**Απόστολος Ξενάκης** Επίκουρος Καθηγητής Τμήμα Ψηφιακών Συστημάτων, Πανεπιστήμιο Θεσσαλίας

Ημερομηνία έγκρισης: 03-10-20024

Περίληψη

Στο πρώτο μέρος της παρούσας εργασίας παρουσιάζεται η ανάπτυξη μιας βιβλιοθήκης για την αυτοματοποιημένη συλλογή, επεξεργασία και οπτικοποίηση μετεωρολογικών δεδομένων. Η βιβλιοθήκη αναπτύχθηκε σε γλώσσα Python με χρήση κατάλληλων μονάδων κώδικα. Μαζί με την παρουσίαση δίνονται και παραδείγματα γραφημάτων χρονοσειρών ή χαρτών που δημιουργήθηκαν με δεδομένα προγνώσεων. Στο δεύτερο μέρος παρουσιάζεται η χρήση πιο προχωρημένων εργαλείων οπτικοποίησης μετεωρολογικών δεδομένων όπως ο συνδυασμός geoserver και opengeoweb.

Abstract

The first part of this paper presents the development of a library for the automated collection, processing and visualization of meteorological data. The library was developed in Python using appropriate code modules. Along with the presentation, examples of time series graphs or maps created with forecast data are given. The second part presents the use of more advanced meteorological data visualization tools such as the combination of geoserver and opengeoweb.

Ευχαριστίες

Θα ήθελα να εκφράσω τις ειλικρινείς μου ευχαριστίες, πρωτίστως προς τη σύζυγό μου, για την αμέριστη υποστήριξη, την αστείρευτη υπομονή και την κατανόηση που επέδειξε καθ' όλη τη διάρκεια των σπουδών μου. Η συμβολή της υπήρξε ανεκτίμητη στην επίτευξη αυτού του στόχου.

Επίσης, θα ήθελα να ευχαριστήσω θερμά τον επιβλέποντα καθηγητή μου, κύριο Ονούφριο Χαραλάμπους, για την εμπιστοσύνη που μου έδειξε, καθώς και για την πολύτιμη καθοδήγηση και βοήθειά του, που συνέβαλαν καθοριστικά στην επιτυχή ολοκλήρωση αυτής της πτυχιακής εργασίας.

ονοματεπώνυμο

ημερομηνία

Περιεχόμενα

1 Εισαγωγή

Η σύγχρονη εξέλιξη της τεχνολογίας έχει επιτρέψει την ανάπτυξη εξειδικευμένων εφαρμογών που αξιοποιούν τα ανοιχτά μετεωρολογικά δεδομένα για τη δημιουργία προγνώσεων και την παροχή σημαντικών πληροφοριών στο κοινό. Στο πλαίσιο αυτό, η παρούσα πτυχιακή εργασία επικεντρώνεται στην ανάπτυξη μιας εφαρμογής που στοχεύει στη συλλογή, επεξεργασία και οπτικοποίηση ανοιχτών μετεωρολογικών προγνώσεων, χρησιμοποιώντας τη γλώσσα προγραμματισμού Python.

Η ανάγκη για πληροφορίες σχετικά με τις μετεωρολογικές συνθήκες έχει ενταθεί στη σύγχρονη κοινωνία, καθώς η κλιματική αλλαγή επηρεάζει ολοένα και περισσότερο τις καθημερινές μας ζωές. Η σωστή πρόβλεψη του καιρού και η κατανόηση των μετεωρολογικών μοντέλων αποτελούν κρίσιμα στοιχεία για τη λήψη αποφάσεων σε πολλούς τομείς, όπως η γεωργία, η ενέργεια, και η ανθρώπινη ασφάλεια.

Η πρόκληση που αντιμετωπίζουμε είναι η ανάπτυξη μιας ευέλικτης και αποδοτικής εφαρμογής που να επιτρέπει την αποτελεσματική συλλογή δεδομένων μετεωρολογικών προγνώσεων από ανοιχτές πηγές, την επεξεργασία αυτών των δεδομένων με ακρίβεια και την ενδελεχή οπτικοποίηση τους για την ευκολότερη κατανόηση.

Στο πλαίσιο της εφαρμογής, θα χρησιμοποιηθούν γραφικά προγράμματα και εργαλεία Python, με έμφαση στις βιβλιοθήκες για την επεξεργασία δεδομένων (όπως το Pandas και το NumPy), τη δημιουργία γραφημάτων (με το Matplotlib) και την αλληλεπίδραση με APIs για τη συλλογή δεδομένων μετεωρολογικών προγνώσεων.

Με την υλοποίηση αυτής της εφαρμογής, επιδιώκουμε όχι μόνο την αποτελεσματική προετοιμασία και παρουσίαση μετεωρολογικών δεδομένων αλλά και τη δημιουργία ενός εργαλείου που θα μπορεί να εξυπηρετεί την κοινότητα και τους ενδιαφερόμενους για την πρόβλεψη των καιρικών συνθηκών με στοιχεία που προέρχονται από αξιόπιστες ανοιχτές πηγές.

2 Βιβλιογραφική ανασκόπηση

Η ανάπτυξη εφαρμογών για τη συλλογή, επεξεργασία και οπτικοποίηση ανοιχτών μετεωρολογικών προγνώσεων αποτελεί πεδίο έρευνας που έχει αναπτυχθεί στα πλαίσια της ψηφιακής εποχής, όπου η πρόσβαση σε δεδομένα και η επεξεργασία τους έχει εξελιχθεί σημαντικά. Οι εφαρμογές αυτές καλύπτουν μια ευρεία γκάμα χρήσεων, από απλές προβλέψεις καιρού μέχρι προηγμένες αναλύσεις κλιματικών μοντέλων.

Η χρήση ανοιχτών μετεωρολογικών δεδομένων έχει αυξηθεί σημαντικά με την εμφάνιση αξιόπιστων και ανοιχτών πηγών πληροφοριών. Οι Μετεωρολογικές Υπηρεσίες, κυβερνητικά όργανα, και διεθνείς οργανισμοί παρέχουν δωρεάν πρόσβαση σε μεγάλο όγκο μετεωρολογικών δεδομένων, ενθαρρύνοντας την ανάπτυξη καινοτόμων εφαρμογών.

Οι εφαρμογές που ασχολούνται με την συλλογή και επεξεργασία μετεωρολογικών δεδομένων χρησιμοποιούν συχνά APIs (Διεπαφές Προγραμματισμού Εφαρμογών) που παρέχουν πρόσβαση σε πραγματικό χρόνο σε δεδομένα καιρού. Παραδείγματα περιλαμβάνουν το OpenWeatherMap API, το NOAA API και το Weatherbit API. Οι αναζητήσεις που γίνονται μέσω αυτών των διεπαφών επιτρέπουν την λήψη αναλυτικών μετεωρολογικών πληροφοριών όπως θερμοκρασία, υγρασία, και ταχύτητα ανέμου.

Η επεξεργασία μετεωρολογικών δεδομένων απαιτεί συχνά τη χρήση εξειδικευμένων βιβλιοθηκών προγραμματισμού. Οι Python βιβλιοθήκες όπως το Pandas και το NumPy προσφέρουν ισχυρά εργαλεία για τη διαχείριση και τον υπολογισμό στατιστικών στοιχείων σε μεγάλα σύνολα δεδομένων. Επίσης, οι βιβλιοθήκες όπως το Matplotlib και το Seaborn χρησιμοποιούνται για τη δημιουργία γραφημάτων και οπτικοποιήσεων που διευκολύνουν την κατανόηση των μετεωρολογικών δεδομένων.

Σε προηγούμενες εργασίες, παρόμοιες εφαρμογές έχουν αναπτυχθεί με επιτυχία για την ανάλυση και πρόβλεψη των καιρικών συνθηκών. Οι αναλύσεις αυτές έχουν εφαρμοστεί σε διάφορους τομείς, όπως η γεωργία, η ενέργεια, και οι υδάτινοι πόροι.

Η παρούσα εργασία συμβάλλει στον επεκτατικό κορμό της βιβλιογραφίας παρέχοντας μια ενδελεχή ανάλυση των προηγούμενων ερευνητικών προσπαθειών στον τομέα, καθώς και προτείνοντας μια νέα προσέγγιση για την ανάπτυξη εφαρμογών με εστίαση στην συλλογή, επεξεργασία και οπτικοποίηση ανοιχτών μετεωρολογικών προγνώσεων με χρήση της γλώσσας προγραμματισμού Python.

3 Ανάπτυξη λογισμικού

Η εφαρμογή αναπτύχθηκε με γνώμονα τη συλλογή, επεξεργασία και οπτικοποίηση μετεωρολογικών δεδομένων, με κύριο ενδιαφέρον τη θερμοκρασία σε συγκεκριμένες γεωγραφικές θέσεις.

3.1 Εισαγωγή στην Python

Η Python είναι μια δυναμική και υψηλού επιπέδου γλώσσα προγραμματισμού, η οποία έχει αποκτήσει μεγάλη δημοτικότητα λόγω της απλότητάς της, της ευκολίας χρήσης της και της ευελιξίας της σε διάφορους τομείς της επιστήμης και της τεχνολογίας. Χρησιμοποιείται ευρέως σε τομείς όπως η ανάπτυξη εφαρμογών ιστού (web development), η επιστημονική ανάλυση δεδομένων (data science), η μηχανική μάθηση (machine learning), η αυτοματοποίηση διαδικασιών, και πολλές άλλες εφαρμογές.

Ένα από τα σημαντικά πλεονεκτήματα της Python είναι η μεγάλη γκάμα διαθέσιμων βιβλιοθηκών και εργαλείων που διευκολύνουν την ανάπτυξη σύνθετων εφαρμογών. Η βιβλιοθήκη της Python περιλαμβάνει έτοιμες συναρτήσεις και μεθόδους για την επίλυση κοινών προβλημάτων προγραμματισμού, μειώνοντας έτσι τον χρόνο και την προσπάθεια που απαιτούνται για την ανάπτυξη λογισμικού. Επιπλέον, η σαφήνεια και η λιτότητα του συντακτικού της καθιστούν την Python ιδανική για τη γρήγορη ανάπτυξη λογισμικού και πρωτοτύπων, επιτρέποντας στους προγραμματιστές να επικεντρωθούν περισσότερο στην επίλυση των προβλημάτων παρά στη γραμμή κώδικα.

3.2 Χρησιμοποιούμενες βιβλιοθήκες

Για την εφαρμογή , χρησιμοποιούνται διάφορες βιβλιοθήκες της Python που καλύπτουν ένα ευρύ φάσμα λειτουργιών, από τη συλλογή δεδομένων μέχρι την επεξεργασία και την οπτικοποίηση αυτών. Αυτές οι βιβλιοθήκες εξασφαλίζουν την ομαλή ροή των δεδομένων και την αποτελεσματική διαχείρισή τους, καθιστώντας δυνατή την υλοποίηση της εφαρμογής με οργανωμένο και δομημένο τρόπο.

reguests: Η βιβλιοθήκη αυτή χρησιμοποιείται για την αποστολή αιτημάτων HTTP. Η βασική λειτουργία της είναι η requests.get(url), η οποία επιστρέφει το περιεχόμενο από τον συγκεκριμένο σύνδεσμο.

BeautifulSoup: Χρησιμοποιείται για την εξαγωγή δεδομένων από HTML σελίδες. Η κεντρική συνάρτηση είναι η BeautifulSoup(html_doc, 'html.parser'), η οποία επιστρέφει ένα αντικείμενο που αντιπροσωπεύει το HTML δέντρο της σελίδας.

Cfgrib: Αυτή η βιβλιοθήκη παρέχει πρόσβαση σε αρχεία GRIB2, τα οποία χρησιμοποιούνται κυρίως για τη μετεωρολογική ανάλυση. Μια βασική μέθοδος είναι η cfgrib.open_datasets('filename.grib') που φορτώνει το αρχείο ως ένα σύνολο δεδομένων.

Numpy: Η βιβλιοθήκη προσφέρει λειτουργίες όπως numpy.array(data) για τη μετατροπή δεδομένων σε πίνακες, ενώ παρέχει και στατιστικές συναρτήσεις όπως numpy.mean(array) για τον υπολογισμό του μέσου όρου.

Matplotlib: Χρησιμοποιείται για τη δημιουργία γραφημάτων και χαρτών. Η βασική συνάρτηση είναι η plt.plot(x, y) που δημιουργεί ένα διάγραμμα γραμμών από τα δεδομένα x και y.

OpenCV: Παρέχει εργαλεία για την επεξεργασία εικόνων, όπως η συνάρτηση cv2.imread('filename') για το άνοιγμα εικόνων και cv2.imshow('image', img) για την προβολή εικόνων.

3.3 Αναλυτική Περιγραφή των Βιβλιοθηκών και της Χρήσης τους

Σε αυτό το κεφάλαιο, θα αναλύσουμε τις βιβλιοθήκες **Python** που χρησιμοποιούνται στην εφαρμογή για τη διαχείριση, επεξεργασία και οπτικοποίηση μετεωρολογικών δεδομένων. Η ανάλυση περιλαμβάνει την περιγραφή των συναρτήσεων, των ορισμάτων τους και των αποτελεσμάτων τους.

3.3.1 Συλλογή δεδομένων

Για την οπτικοποίηση των μετεωρολογικών δεδομένων χρησιμοποιήσαμε τις εξής βιβλιοθήκες Η συλλογή των δεδομένων γίνεται με τη χρήση των βιβλιοθηκών **requests**, **BeautifulSoup** και **bz2**, οι οποίες συνεργάζονται για τη λήψη και αποσυμπίεση μετεωρολογικών αρχείων από το διαδίκτυο.

3.3.2 Συνάρτηση download_files

Η συνάρτηση αυτή χρησιμοποιείται για να κατεβάσει και να αποσυμπιέσει αρχεία GRIB2 που είναι διαθέσιμα σε μια διαδικτυακή διεύθυνση.

Ορίσματα (Inputs):

- **directory_url (str):** Η URL διεύθυνση του φακέλου που περιέχει τα αρχεία GRIB2.
- **base_directory (str):** Η τοπική διαδρομή (path) όπου θα αποθηκευτούν τα κατεβασμένα και αποσυμπιεσμένα αρχεία.

Έξοδος (Output):

- **files_path (str):** Η τοπική διαδρομή (path) όπου αποθηκεύονται τα αποσυμπιεσμένα αρχεία.

Η συνάρτηση αρχικά στέλνει ένα αίτημα **HTTP** μέσω της **requests.get** για να λάβει το περιεχόμενο της σελίδας που περιέχει τα αρχεία **GRIB2**. Στη συνέχεια, χρησιμοποιεί τη **BeautifulSoup** για να εξάγει τους συνδέσμους των αρχείων από το **HTML** της σελίδας. Κάθε αρχείο κατεβαίνει και αποσυμπιέζεται χρησιμοποιώντας τη βιβλιοθήκη **bz2**, πριν αποθηκευτεί στον τοπικό φάκελο που καθορίζεται από την παράμετρο **base_directory.**

3.3.3 Επεξεργασία δεδομένων

Η επεξεργασία των μετεωρολογικών δεδομένων πραγματοποιείται μέσω των βιβλιοθηκών **cfgrib** και **numpy**, οι οποίες επιτρέπουν την ανάγνωση των αρχείων **GRIB2** και την επεξεργασία των δεδομένων που περιέχουν.

3.3.4 Συνάρτηση get_variable

Ορίσματα (Inputs):

- **file_path (str):** Η τοπική διαδρομή (path) όπου βρίσκονται τα αρχεία GRIB2.
- **variable_name (str):** Το όνομα της μεταβλητής που θέλουμε να εξάγουμε (π.χ., "t2m" για θερμοκρασία).
- **latitude (float):** Το γεωγραφικό πλάτος (latitude) του σημείου για το οποίο θέλουμε την τιμή της μεταβλητής.
- **longitude (float):** Το γεωγραφικό μήκος (longitude) του σημείου για το οποίο θέλουμε την τιμή της μεταβλητής.

Έξοδος (Output):

- **hour_list (list of str):** Λίστα με τις ώρες για τις οποίες εξήχθησαν τα δεδομένα από τα αρχεία GRIB2.
- **variable_list (list of float):** Λίστα με τις τιμές της μεταβλητής για τις αντίστοιχες ώρες

Η συνάρτηση διαβάζει τα αρχεία **GRIB2** που περιέχονται στη διαδρομή **file_path** και εξάγει τη μεταβλητή που καθορίζεται από το **variable_name** για το σημείο που προσδιορίζεται από τις συντεταγμένες **latitude** και **longitude**. Η τιμή της μεταβλητής υπολογίζεται μέσω παρεμβολής χρησιμοποιώντας τη **RegularGridInterpolator** από τη **scipy.interpolate**.

3.3.5 Οπτικοποίηση Δεδομένων

Η οπτικοποίηση των δεδομένων γίνεται μέσω των βιβλιοθηκών **matplotlib** και **opencv**. Οι βιβλιοθήκες αυτές χρησιμοποιούνται για τη δημιουργία γραφημάτων και βίντεο από τα μετεωρολογικά δεδομένα.

3.3.6 Συνάρτηση visualize_temperature

Η συνάρτηση αυτή δημιουργεί έναν θερμικό χάρτη που απεικονίζει την κατανομή της θερμοκρασίας σε μια γεωγραφική περιοχή.

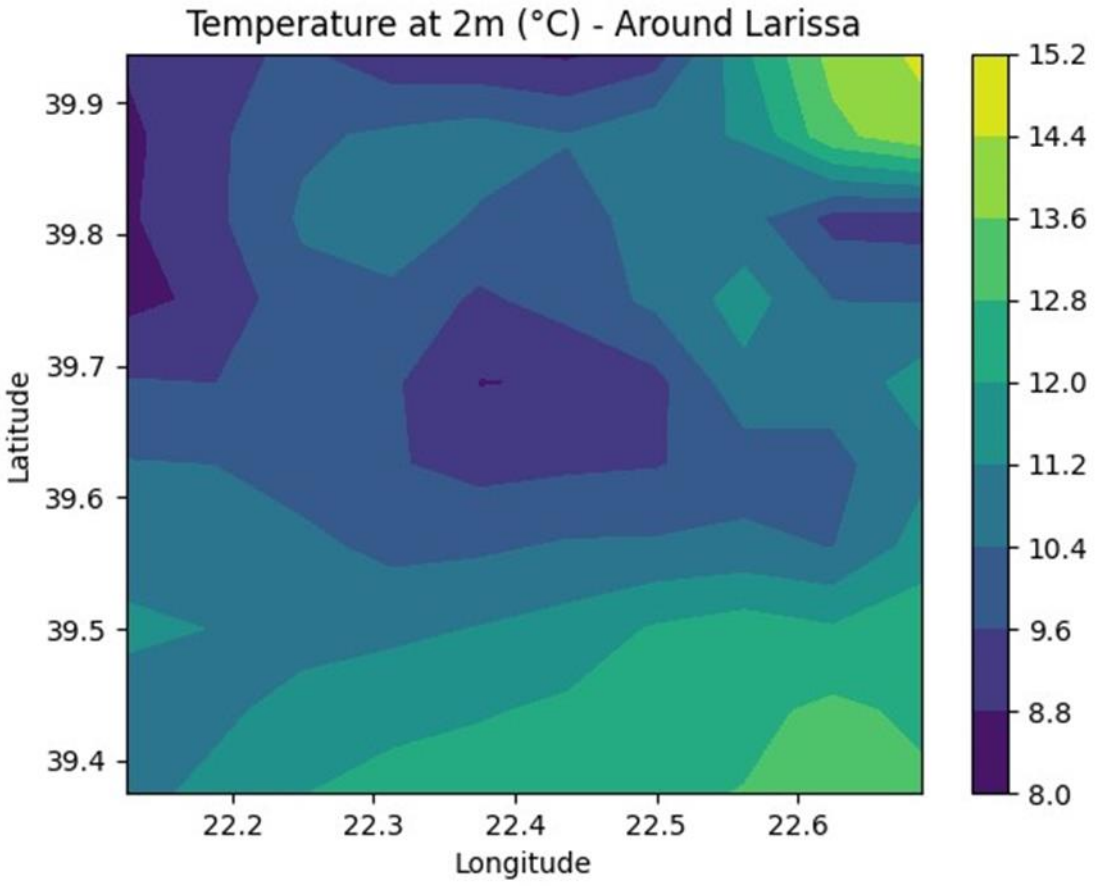
Ορίσματα (Inputs):

- **latitudes (numpy.ndarray):** Πίνακας που περιέχει τις γεωγραφικές συντεταγμένες πλάτους (latitude).
- **longitudes (numpy.ndarray):** Πίνακας που περιέχει τις γεωγραφικές συντεταγμένες μήκους (longitude).
- **temperature_data (numpy.ndarray):** Πίνακας που περιέχει τις τιμές της θερμοκρασίας σε Kelvin.
- **save_filename (str, προαιρετικό):** Το όνομα του αρχείου στο οποίο θα αποθηκευτεί το γράφημα. Αν δεν δοθεί, το γράφημα δεν αποθηκεύεται αλλά εμφανίζεται στην οθόνη.
- **hour (str, προαιρετικό):** Η ώρα για την οποία δημιουργείται ο χάρτης.
- **target_lat (float, προαιρετικό):** Το γεωγραφικό πλάτος του κεντρικού σημείου ενδιαφέροντος.

- **target_lon (float, προαιρετικό):** Το γεωγραφικό μήκος του κεντρικού σημείου ενδιαφέροντος.

Έξοδος (Output):

Δεν υπάρχει άμεση έξοδος. Η συνάρτηση δημιουργεί και αποθηκεύει (ή εμφανίζει) έναν θερμικό χάρτη θερμοκρασίας. Δημιουργεί έναν θερμικό χάρτη χρησιμοποιώντας τη **matplotlib** και την **plt.contourf()** για την απεικόνιση της θερμοκρασίας. Οι τιμές της θερμοκρασίας μετατρέπονται από Kelvin σε Κελσίου και στη συνέχεια απεικονίζονται με χρωματική κλίμακα. Εάν καθοριστεί η παράμετρος **save_filename**, ο χάρτης αποθηκεύεται ως αρχείο εικόνας, διαφορετικά εμφανίζεται στην οθόνη.



Εικόνα 1: Συνάρτηση visualize_temperature

3.3.7 Συνάρτηση create_video

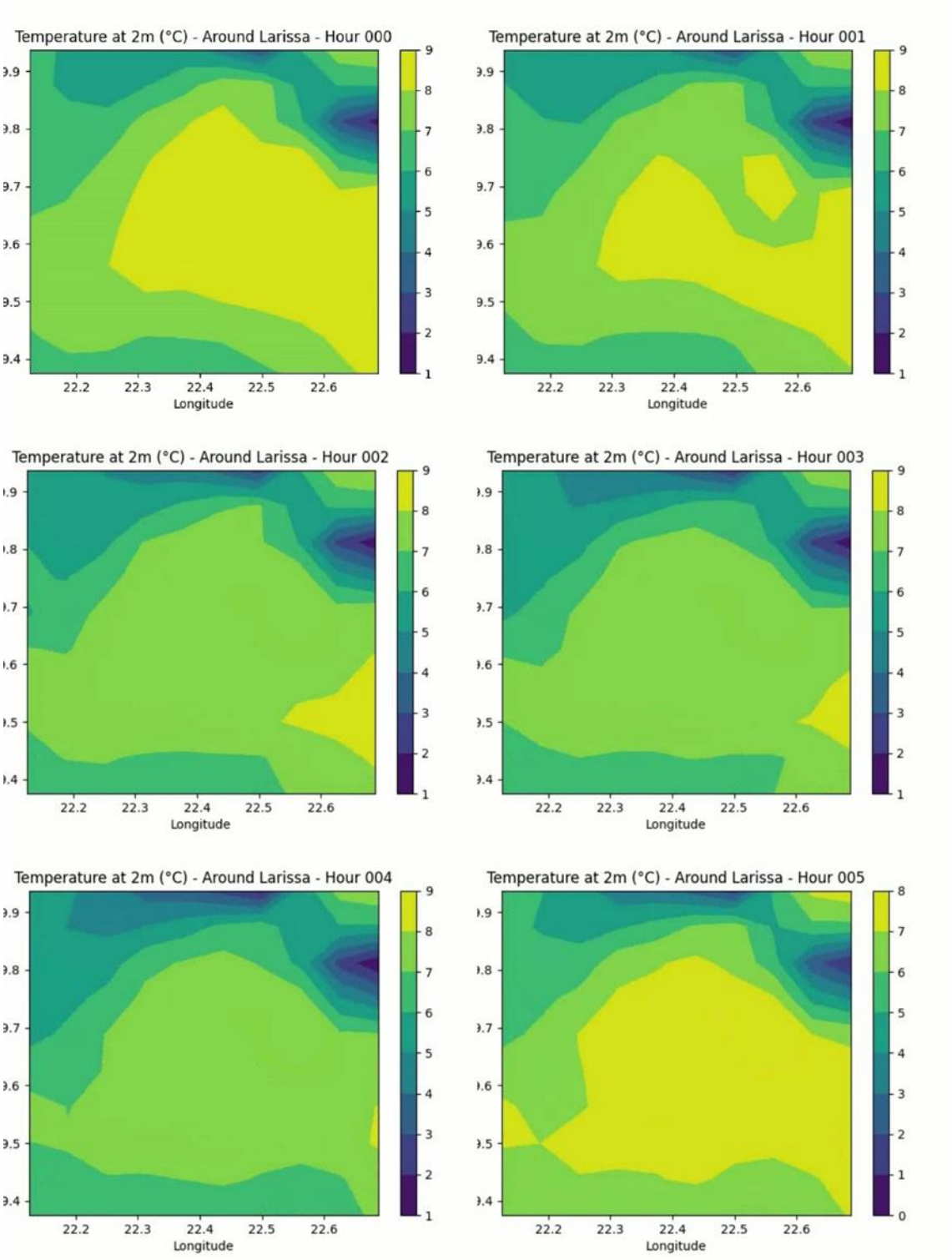
Δημιουργεί ένα βίντεο από μια ακολουθία εικόνων που απεικονίζουν την εξέλιξη της θερμοκρασίας με την πάροδο του χρόνου.

Ορίσματα (Inputs):

- **files_path (str):** Η τοπική διαδρομή (path) όπου βρίσκονται οι εικόνες που θα χρησιμοποιηθούν για τη δημιουργία του βίντεο.
- **video_output_path (str):** Η τοπική διαδρομή (path) και το όνομα του αρχείου στο οποίο θα αποθηκευτεί το παραγόμενο βίντεο.

Έξοδος (Output):

Η **create_video** χρησιμοποιεί τη βιβλιοθήκη **opencv** για να διαβάσει τις εικόνες από τη διαδρομή **files_path** και να τις συνθέσει σε ένα βίντεο. Το βίντεο αποθηκεύεται ως αρχείο .mp4 στη διαδρομή που καθορίζεται από το **video_output_path**.



Εικόνα 2: Συνάρτηση create_video

3.3.8 Συνάρτηση process_weather_data

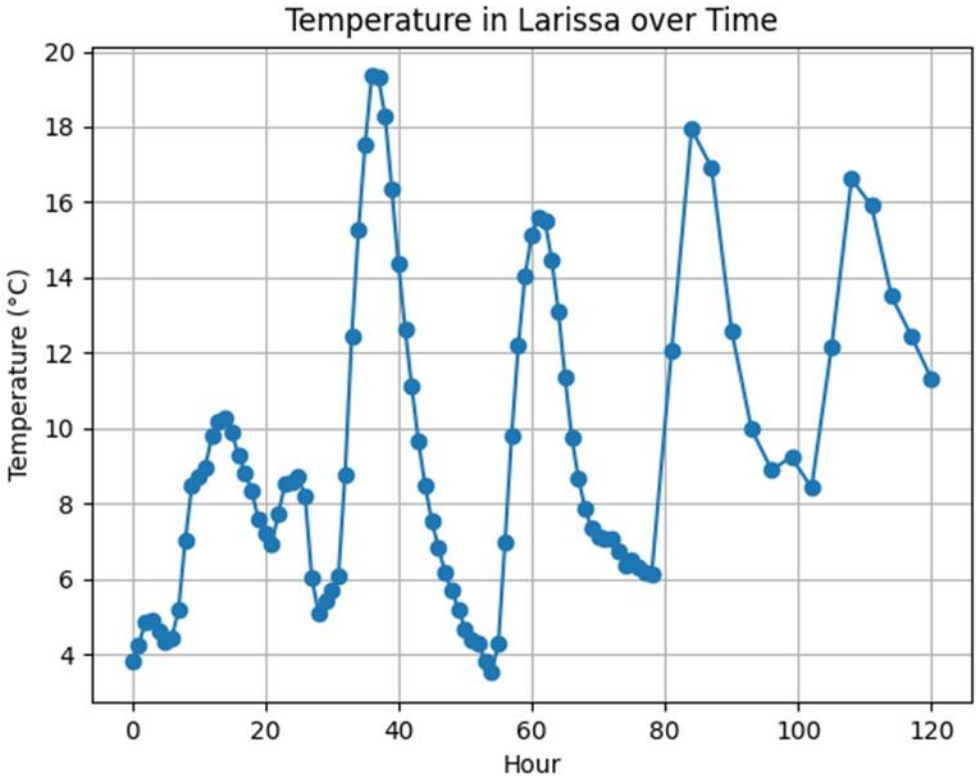
Εκτελεί την κεντρική διαδικασία της συλλογής, επεξεργασίας και οπτικοποίησης των μετεωρολογικών δεδομένων.

Ορίσματα (Inputs):

- **data_directory_url (str):** Η URL διεύθυνση του φακέλου που περιέχει τα αρχεία GRIB2.
- **base_directory (str):** Η τοπική διαδρομή (path) όπου θα αποθηκευτούν τα κατεβασμένα και αποσυμπιεσμένα αρχεία.
- **variable_name (str):** Το όνομα της μεταβλητής που θέλουμε να εξάγουμε (π.χ., "t2m" για θερμοκρασία).
- **latitude (float):** Το γεωγραφικό πλάτος (latitude) του σημείου για το οποίο θέλουμε την τιμή της μεταβλητής.
- **longitude (float):** Το γεωγραφικό μήκος (longitude) του σημείου για το οποίο θέλουμε την τιμή της μεταβλητής.

Έξοδος (Output):

Η **process_weather_data** καλεί διαδοχικά τις συναρτήσεις **download_files**, **get_variable**, **visualize_temperature,** και **create_video**, για να ολοκληρώσει την πλήρη διαδικασία από τη συλλογή έως την οπτικοποίηση των μετεωρολογικών δεδομένων.



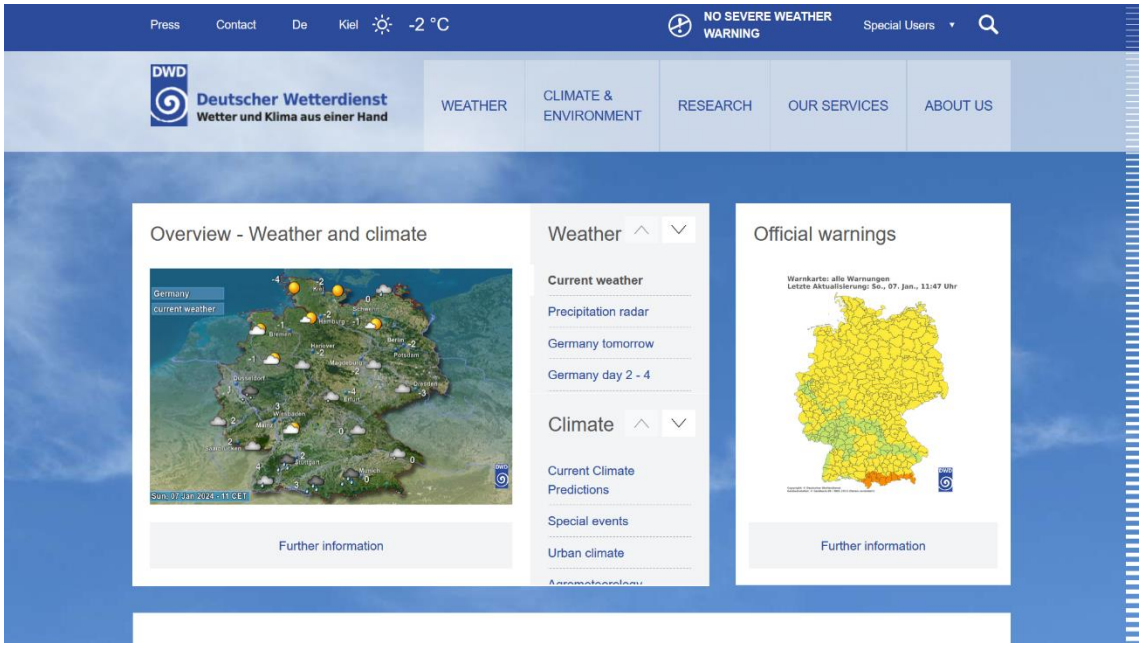
Εικόνα 3: Συνάρτηση process_weather_data

matplotlib: Χρησιμοποιείται για τη δημιουργία γραφημάτων και χαρτών.

opencv: Χρησιμοποιείται για τη δημιουργία εικόνων και τον χειρισμό βίντεο.

3.4 Πηγή Δεδομένων

Η πηγή των opendata μετεωρολογικών δεδομένων είναι ο Γερμανικός Υπηρεσίας Καιρού (DWD - Deutscher Wetterdienst) Εικόνα 1 ο οποίος αντιπροσωπεύει μία από τις προηγμένες μετεωρολογικές υπηρεσίες παγκοσμίως, προσφέροντας υψηλής ποιότητας πληροφορίες για τον καιρό, το κλίμα και άλλα μετεωρολογικά φαινόμενα. Η DWD αναλαμβάνει το ρόλο του επιστημονικού και τεχνικού φορέα για τη μετεωρολογική παρακολούθηση, την εκπόνηση προγνώσεων και την παροχή μετεωρολογικών υπηρεσιών στη Γερμανία. Ένα σημαντικό κομμάτι της προσφοράς της DWD είναι τα ανοικτά μετεωρολογικά δεδομένα (opendata), τα οποία αντιπροσωπεύουν μια πολύτιμη πηγή πληροφοριών για ερευνητές, προγραμματιστές, και το ευρύ κοινό. Τα μετεωρολογικά αρχεία που παρέχονται υπό την ετικέτα opendata είναι διαθέσιμα δωρεάν και ανοιχτά για το κοινό, συμβάλλοντας στη διευκόλυνση της πρόσβασης σε αξιόπιστες μετεωρολογικές πληροφορίες.



Εικόνα 4: website dwd

4 Προχωρημένα Εργαλεία Οπτικοποίησης

Σε αυτό το κεφάλαιο θα παρουσιάσουμε τη διαδικασία χρήσης προηγμένων εργαλείων για την εισαγωγή, διαχείριση και οπτικοποίηση μετεωρολογικών δεδομένων. Εστιάζουμε στην εφαρμογή δύο βασικών τεχνολογιών: του GeoServer για τη δημοσίευση και διαχείριση γεωχωρικών δεδομένων, και του OpenGeoWeb για την προβολή και αλληλεπίδραση με τα δεδομένα μέσω ενός διαδικτυακού περιβάλλοντος. Ακολουθώντας αυτή τη διαδικασία, θα εξασφαλίσουμε την ορθή ενσωμάτωση και παρουσίαση μεγάλων γεωχωρικών συνόλων δεδομένων, όπως αυτά που αφορούν τη μετεωρολογία.

4.1 GeoServer και Εισαγωγή Μετεωρολογικών Δεδομένων

Ο GeoServer είναι μια ισχυρή πλατφόρμα ανοιχτού κώδικα, σχεδιασμένη για την κοινή χρήση, επεξεργασία και προβολή γεωχωρικών δεδομένων μέσω του διαδικτύου. Υποστηρίζει διεθνή πρότυπα όπως το Web Map Service (WMS), το Web Feature Service (WFS) και το Web Coverage Service (WCS), τα οποία διευκολύνουν την ανταλλαγή δεδομένων ανάμεσα σε διαφορετικά συστήματα και εφαρμογές. Ο GeoServer μπορεί να χειριστεί μια ευρεία ποικιλία μορφών γεωχωρικών δεδομένων, όπως Shapefiles, GeoTIFFs, και δεδομένα από βάσεις όπως το PostGIS, επιτρέποντας τη δημοσίευση και διαχείριση αυτών των δεδομένων μέσω του διαδικτύου.

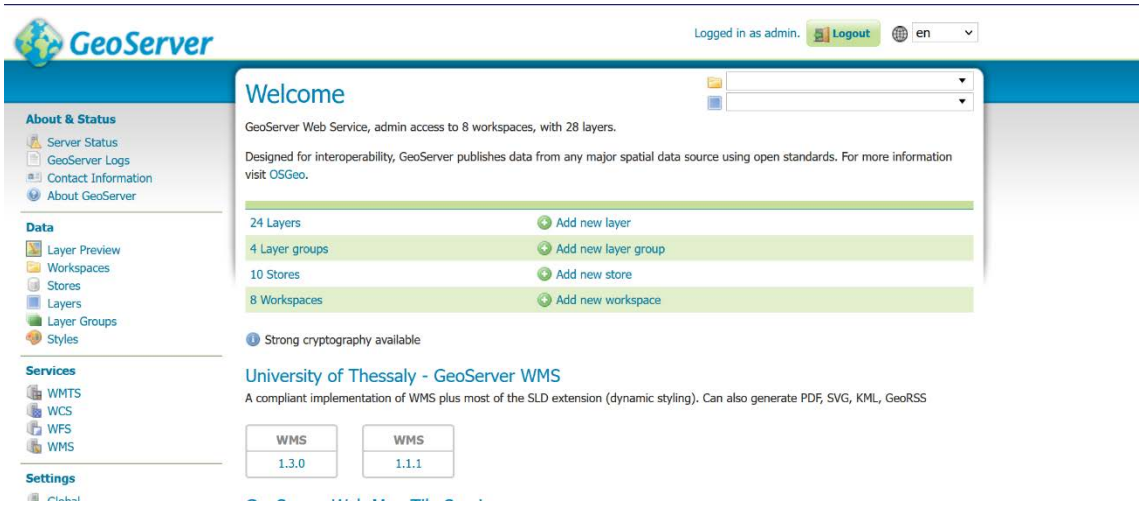
4.1.1 Γενική Περιγραφή του GeoServer

Η βασική λειτουργικότητα του GeoServer περιλαμβάνει τα εξής:

- **Δημοσίευση Χαρτών**: Παρέχει τη δυνατότητα δημοσίευσης γεωγραφικών δεδομένων με τη μορφή χαρτών, οι οποίοι μπορούν να προβληθούν και να χρησιμοποιηθούν από άλλες εφαρμογές GIS.
- **Προβολή Δεδομένων**: Προσφέρει διαδραστική προβολή των γεωχωρικών δεδομένων μέσω διαδικτυακών υπηρεσιών όπως WMS, WFS, και WCS.

- **Διαχείριση και Επεξεργασία Δεδομένων**: Ο GeoServer παρέχει εργαλεία για τη διαχείριση, επεξεργασία και προσαρμογή των δεδομένων, με δυνατότητα εφαρμογής διαφόρων στυλ και κανόνων απεικόνισης.

Ο GeoServer είναι ιδανικός για τη διαχείριση μεγάλων γεωχωρικών συνόλων δεδομένων, όπως τα μετεωρολογικά δεδομένα, επιτρέποντας την οπτικοποίησή τους μέσω ενός διαδικτυακού περιβάλλοντος.



Εικόνα 5: website Geoserver

4.1.2 Διαδικασία Εισαγωγής Δεδομένων στον GeoServer

Για την εισαγωγή των επεξεργασμένων μετεωρολογικών δεδομένων στον GeoServer, επιλέξαμε να χρησιμοποιήσουμε τη μορφή ImageMosaic. Αυτή η μορφή επιτρέπει την αποθήκευση πολλαπλών εικόνων, όπως GeoTIFFs, ως ενιαίο γεωχωρικό επίπεδο (layer) στον GeoServer, διευκολύνοντας την προβολή χρονοσειρών δεδομένων. Ειδικότερα, τα δεδομένα θερμοκρασίας που εξάχθηκαν από αρχεία GRIB2, μετατράπηκαν σε GeoTIFFs, και στη συνέχεια εισήχθησαν στον GeoServer ως ImageMosaic.

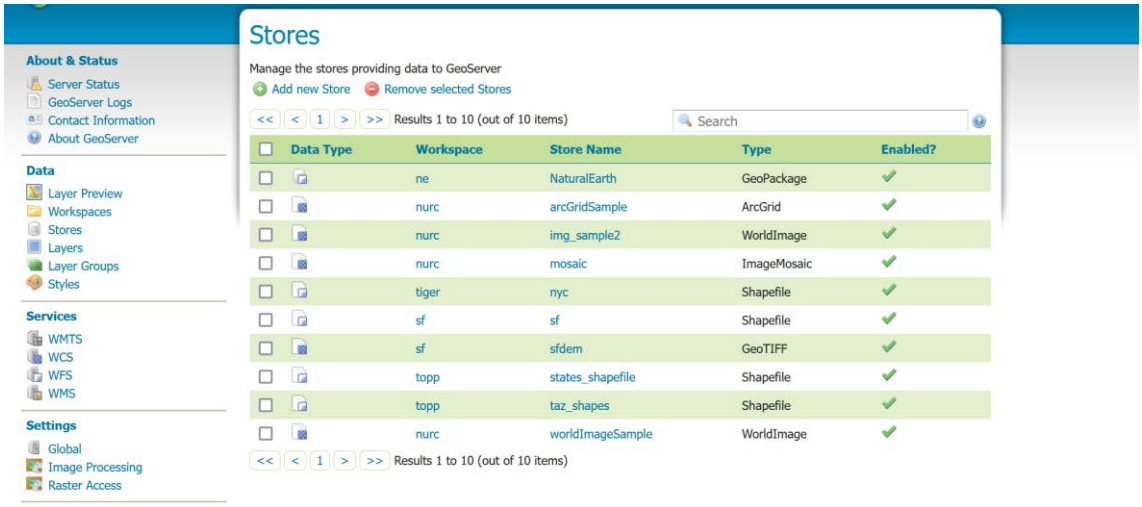
4.1.3 Προετοιμασία των Δεδομένων

Πριν από την εισαγωγή των δεδομένων στον GeoServer, είναι απαραίτητο να τα μετατρέψουμε σε μορφή συμβατή με το ImageMosaic. Στην περίπτωση μας, τα δεδομένα θερμοκρασίας αποθηκεύτηκαν ως GeoTIFFs, κάθε ένα από τα οποία αντιπροσωπεύει τη θερμοκρασία για μία συγκεκριμένη χρονική στιγμή.

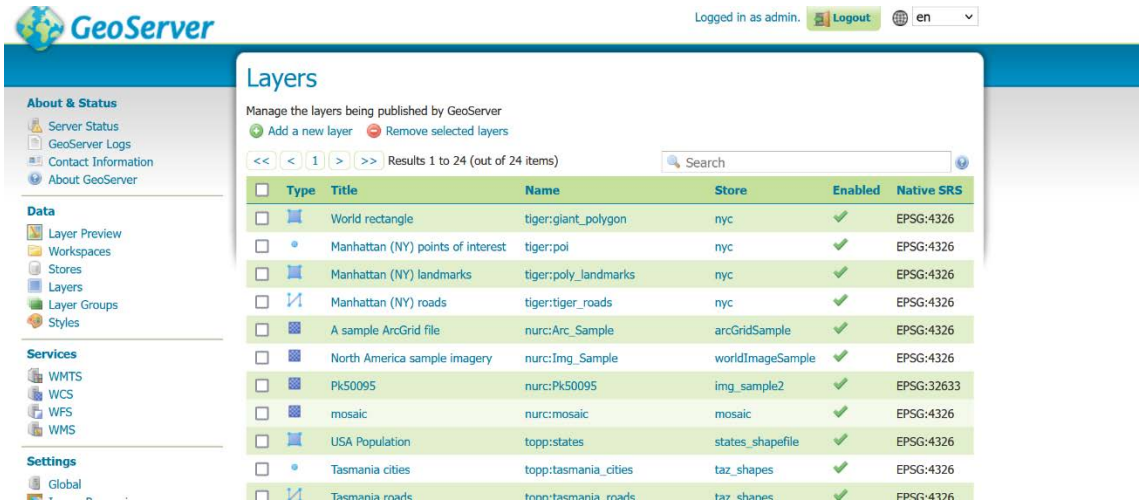
4.1.4 Δημιουργία ImageMosaic στον GeoServer

Η διαδικασία δημιουργίας ενός ImageMosaic στο GeoServer περιλαμβάνει τα εξής βήματα:

- **Δημιουργία Καταλόγου Δεδομένων**: Στον κατάλογο δεδομένων του GeoServer δημιουργούμε έναν φάκελο που περιέχει όλα τα GeoTIFFs. Κάθε αρχείο TIFF αναπαριστά δεδομένα θερμοκρασίας για μια συγκεκριμένη χρονική στιγμή.
- **Δημιουργία ενός νέου Store**: Στη διεπαφή του GeoServer επιλέγουμε τη δημιουργία ενός νέου Store και ορίζουμε τη μορφή ImageMosaic, επιλέγοντας τον φάκελο που περιέχει τα GeoTIFFs ως πηγή δεδομένων.
- **Διαμόρφωση και Δημοσίευση Layer**: Αφού δημιουργηθεί το Store, το GeoServer αυτόματα αναγνωρίζει τα GeoTIFFs και δημιουργεί το Layer. Στη συνέχεια, μπορούμε να προσαρμόσουμε τις παραμέτρους προβολής και το στυλ, και να δημοσιεύσουμε το Layer για χρήση μέσω του διαδικτύου.



Εικόνα 6: website Geoserver Stores



Εικόνα 7: website Geoserver Layers

4.1.5 Πλεονεκτήματα της Χρήσης ImageMosaic

Η χρήση του ImageMosaic στον GeoServer παρέχει σημαντικά πλεονεκτήματα:

- **Διαχείριση Χρονοσειρών**: Διευκολύνει την προβολή και ανάλυση της εξέλιξης των μετεωρολογικών φαινομένων σε διαφορετικές χρονικές στιγμές.
- **Βελτιστοποίηση Επεξεργασίας**: Μειώνει τον φόρτο στο σύστημα και βελτιστοποιεί την επεξεργασία μεγάλων συνόλων δεδομένων.
- **Δυνατότητα Κλιμάκωσης**: Η μορφή ImageMosaic επιτρέπει την εύκολη προσθήκη νέων δεδομένων στο υπάρχον Layer, διευκολύνοντας την συνεχή ενημέρωση της βάσης δεδομένων.

Με τη χρήση του GeoServer και της μορφής ImageMosaic, τα επεξεργασμένα μετεωρολογικά δεδομένα εισάγονται επιτυχώς και είναι έτοιμα για προβολή και ανάλυση μέσω διαδικτύου.

4.2 Χρήση του OpenGeoWeb για την Οπτικοποίηση των Δεδομένων

Το OpenGeoWeb είναι μια πλατφόρμα ανοιχτού κώδικα που επιτρέπει την προβολή και ανάλυση γεωχωρικών δεδομένων μέσω του διαδικτύου. Υποστηρίζει πολλαπλά επίπεδα δεδομένων και προσφέρει διεπαφές για τη σύνδεση με γεωχωρικούς διακομιστές, όπως ο GeoServer, μέσω προτύπων όπως το WMS, WFS και WCS. Το OpenGeoWeb είναι σχεδιασμένο για να παρέχει προηγμένα εργαλεία για την αλληλεπίδραση και ανάλυση γεωχωρικών δεδομένων.

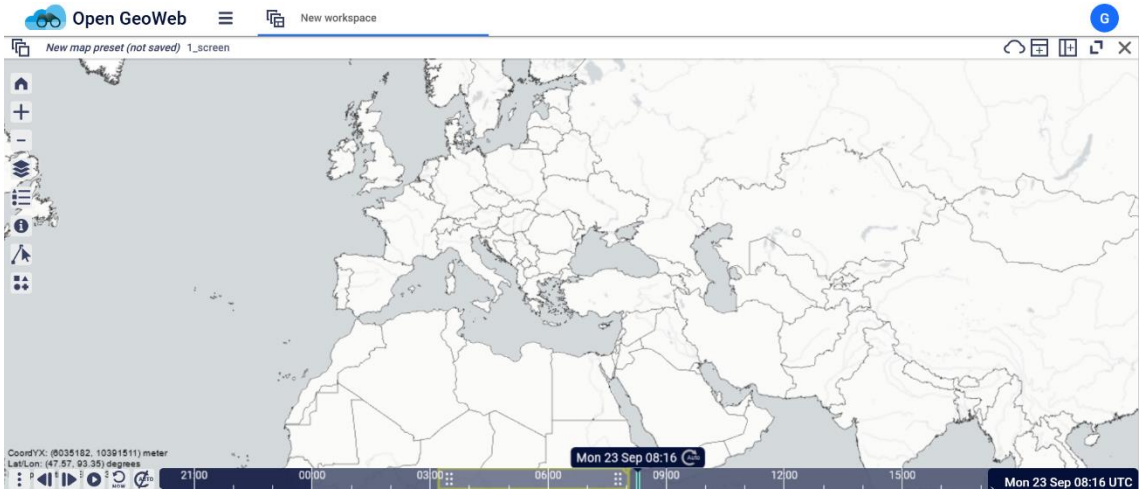
4.2.1 Γενικές Πληροφορίες για το OpenGeoWeb

Τα κύρια χαρακτηριστικά του OpenGeoWeb περιλαμβάνουν:

- **Υποστήριξη Πολλαπλών Επιπέδων Δεδομένων**: Δυνατότητα προβολής και ανάλυσης πολλαπλών layers ταυτόχρονα, ενσωματώνοντας δεδομένα από διάφορες πηγές.
- **Διεπαφές Διασύνδεσης με Διακομιστές Γεωχωρικών Δεδομένων**: Δυνατότητα σύνδεσης με διακομιστές γεωχωρικών δεδομένων μέσω προτύπων όπως το WMS και το WFS.

- **Εργαλεία Αλληλεπίδρασης και Ανάλυσης**: Παρέχει εργαλεία για δυναμική αλληλεπίδραση με τα δεδομένα, όπως επιλογή περιοχών ενδιαφέροντος και προβολή χρονοσειρών.

Το OpenGeoWeb είναι ιδανικό για τη διάθεση γεωχωρικών δεδομένων σε χρήστες μέσω ενός διαδικτυακού περιβάλλοντος, επιτρέποντας την εύκολη πρόσβαση και ανάλυση των δεδομένων.



Εικόνα 8: website Open Geoweb

4.2.2 Σύνδεση του OpenGeoWeb με τον GeoServer

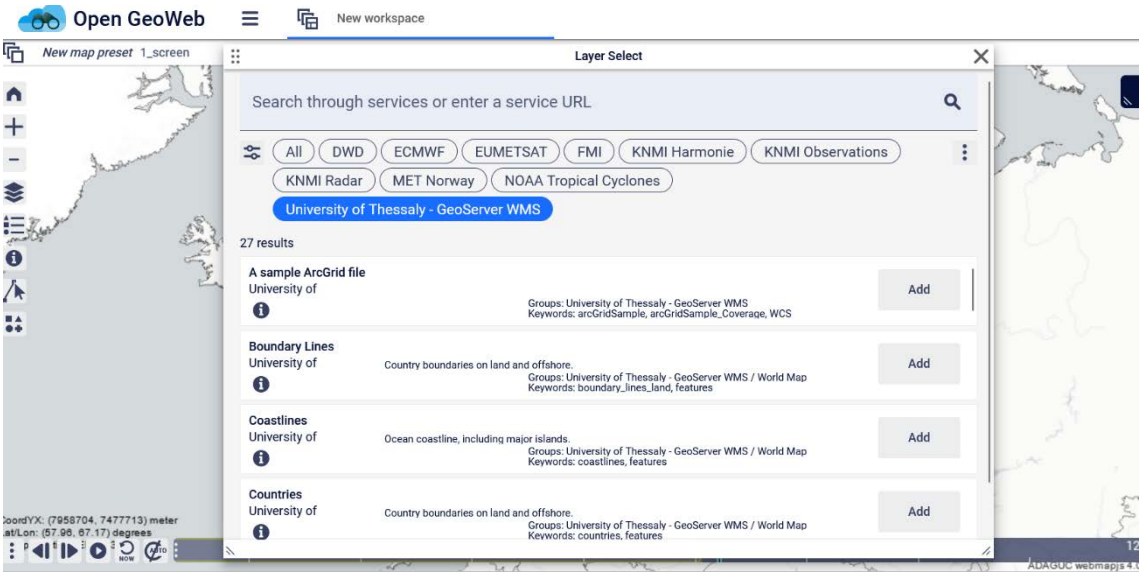
Η σύνδεση του OpenGeoWeb με τον GeoServer γίνεται εύκολα μέσω των προτύπων WMS και WFS. Αυτά τα πρότυπα επιτρέπουν την ανάκτηση και προβολή γεωχωρικών δεδομένων που έχουν δημοσιευτεί στον GeoServer.

4.2.3 Διαδικασία Σύνδεσης του OpenGeoWeb με τον GeoServer

Η σύνδεση μεταξύ του OpenGeoWeb και του GeoServer επιτυγχάνεται μέσω των εξής βημάτων:

- **Προσθήκη του WMS του GeoServer**: Στο περιβάλλον διαχείρισης του OpenGeoWeb, εισάγουμε τη διεύθυνση URL της υπηρεσίας WMS του GeoServer.
- **Επιλογή των Layers για Προβολή**: Μετά τη σύνδεση, το OpenGeoWeb εμφανίζει όλα τα διαθέσιμα layers από τον GeoServer. Επιλέγουμε τα layers που θέλουμε να προβάλουμε.

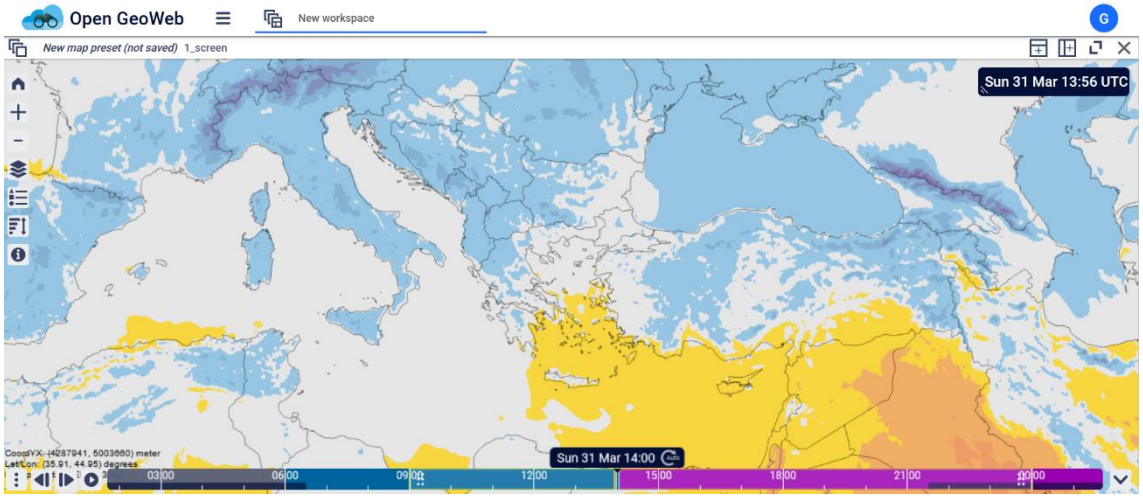
- **Ρύθμιση των Εργαλείων Προβολής**: Μπορούμε να προσαρμόσουμε τα εργαλεία προβολής, όπως το εύρος χρονοσειρών, ζουμ, και επιλογή περιοχών.



Εικόνα 9: Διασύνδεση Open Geoweb με Geoserver UTH

4.2.4 Δυναμική Αλληλεπίδραση με τα Δεδομένα

Αφού συνδεθούν τα δεδομένα, το OpenGeoWeb προσφέρει τη δυνατότητα δυναμικής αλληλεπίδρασης με τα μετεωρολογικά δεδομένα. Οι χρήστες μπορούν να επιλέξουν συγκεκριμένες χρονικές στιγμές, να προβάλουν τις μεταβολές θερμοκρασίας και να αναλύσουν τις χωρικές σχέσεις μεταξύ των δεδομένων. Με το OpenGeoWeb, μπορούμε να επεξεργαστούμε και να προβάλλουμε μετεωρολογικά δεδομένα σε πραγματικό χρόνο. Η πλατφόρμα επιτρέπει την παρακολούθηση και ανάλυση της εξέλιξης των φαινομένων, δίνοντας τη δυνατότητα στους χρήστες να βλέπουν τις αλλαγές στη θερμοκρασία και άλλα μετεωρολογικά φαινόμενα μέσω διαδραστικών χαρτών.



Εικόνα 10: Οπτικοποίηση δεδομένων Geoserver στο OpenGeoweb

5 Συμπεράσματα

Η πτυχιακή εργασία επικεντρώθηκε στην ανάπτυξη μιας εφαρμογής για τη συλλογή, επεξεργασία και οπτικοποίηση μετεωρολογικών δεδομένων, με ιδιαίτερη έμφαση στην θερμοκρασία σε συγκεκριμένες γεωγραφικές περιοχές. Η χρήση της γλώσσας Python υπήρξε κρίσιμη, καθώς η απλότητα και η ευελιξία της, σε συνδυασμό με τις ισχυρές βιβλιοθήκες της, μας επέτρεψαν να επιτύχουμε τα επιθυμητά αποτελέσματα με σχετικά μικρή προσπάθεια. Οι βιβλιοθήκες όπως requests, BeautifulSoup, cfgrib, numpy, matplotlib και OpenCV διαδραμάτισαν καίριο ρόλο στη διαδικασία, παρέχοντας τα απαραίτητα εργαλεία για την ολοκλήρωση της εφαρμογής.

Αρχικά, η διαδικασία συλλογής δεδομένων μέσω των βιβλιοθηκών requests και BeautifulSoup, καθώς και η αποσυμπίεση αρχείων με bz2, ήταν αποτελεσματική για την απόκτηση των απαραίτητων μετεωρολογικών αρχείων GRIB2. Ωστόσο, η συνεργασία των βιβλιοθηκών αυτών παρουσίασε προκλήσεις, κυρίως λόγω της ανάγκης για σωστή ανάλυση των HTML εγγράφων και της διαχείρισης των δεδομένων μεγάλου όγκου.

Η επεξεργασία των δεδομένων με τις βιβλιοθήκες cfgrib και numpy αποδείχτηκε πολύτιμη, διευκολύνοντας την ανάγνωση και ανάλυση των μετεωρολογικών δεδομένων. Οι προκλήσεις που αντιμετωπίστηκαν περιλάμβαναν την ακριβή εξαγωγή μεταβλητών από τα αρχεία GRIB2 και την αντιμετώπιση προβλημάτων με την ενοποίηση των δεδομένων.

Η οπτικοποίηση, μέσω των βιβλιοθηκών matplotlib και OpenCV, παρείχε σημαντικά αποτελέσματα, δημιουργώντας γραφήματα και βίντεο που απεικονίζουν την εξέλιξη της θερμοκρασίας. Αν και η διαδικασία αυτή ήταν σύνθετη, η τελική παρουσίαση των δεδομένων ήταν ικανοποιητική και χρήσιμη για τη κατανόηση των μετεωρολογικών φαινομένων.

Η εργασία αποτέλεσε μία ενδιαφέρουσα πρόκληση, καθώς απαιτούσε συνδυασμένη χρήση πολλών εργαλείων και τεχνικών. Αν μπορούσα να ξανακάνω την πτυχιακή εργασία, πιθανόν θα εστίαζα περισσότερο στη βελτίωση της απόδοσης της επεξεργασίας δεδομένων και στην ενσωμάτωσή τους σε ένα πιο ολοκληρωμένο σύστημα οπτικοποίησης. Η εργασία προσέφερε σημαντικές γνώσεις και εμπειρίες, ενισχύοντας τη κατανόησή μου

για την ανάπτυξη εφαρμογών που χειρίζονται μεγάλες ποσότητες δεδομένων και τη δημιουργία γραφικών απεικονίσεων.

Η συνέχιση της πτυχιακής θα μπορούσε να περιλαμβάνει την ενσωμάτωσή της με άλλες πηγές δεδομένων και τη βελτίωση της διεπαφής χρήστη. Επιπλέον, θα μπορούσε να επεκταθεί σε άλλες μετεωρολογικές παραμέτρους και να συμπεριλάβει πιο προηγμένες τεχνικές ανάλυσης και οπτικοποίησης, όπως η δυναμική ανάλυση χρονοσειρών και η ενσωμάτωσή της με πλατφόρμες ανάλυσης δεδομένων στο διαδίκτυο

Βιβλιογραφία

[1] GeoServer Project. (2023). GeoServer User Manual. Retrieved from https://docs.geoserver.org/stable/en/user/.

[2] Hunter, J. D., Droettboom, M., & Firing, E. (2023). Matplotlib Documentation. Retrieved from https://matplotlib.org/stable/index.html.

[3] Reitz, K. (2023). Requests: HTTP for Humans. Retrieved from https://docs.python-requests.org/en/latest/

[4] European Centre for Medium-Range Weather Forecasts (ECMWF). (2023). GRIB Documentation: A Practical Guide to the Use of GRIB2. Retrieved from https://apps.ecmwf.int/codes/grib/

[5] Deutscher Wetterdienst (DWD). (2023). Open Data. Retrieved from https://www.dwd.de/EN/climate_environment/cdc/cdc.html

Παράρτημα Α

Η συνάρτηση download_files.

```
def download_files(directory_url, base_directory):
    files_path = os.path.join(base_directory,
os.path.basename(directory_url.rstrip('/')))

    if not os.path.exists(files_path):
        os.makedirs(files_path)

    response = requests.get(directory_url)
    downloaded_files_count = 0

    if response.status_code == 200:
        soup = BeautifulSoup(response.text, 'html.parser')
        links = [a['href'] for a in soup.find_all('a')]

        print(f"Files in '{os.path.basename(files_path)}' are being checked for
download...")

        for link in links:
            if link.endswith('.grib2.bz2'):
                file_url = directory_url + link
                current_file_date = get_date_from_filename(link)

                file_path = os.path.join(files_path, link[:-4])

                if not os.path.exists(file_path):
                    response_file = requests.get(file_url)

                    if response_file.status_code == 200:
                        decompressed_content = bz2.decompress(response_file.content)

                        with open(file_path, "wb") as file:
                            file.write(decompressed_content)

                        downloaded_files_count += 1
                        print(f"Downloaded: {link}")
                    else:
                        print(f"Failed to download the file from the URL: {file_url}")

        if downloaded_files_count > 0:
            print(f"Finished checking. Downloaded and saved {downloaded_files_count}
decompressed files in the '{os.path.basename(files_path)}' directory.")

    return files_path
```

Η συνάρτηση visualize_temperature.

```
def visualize_temperature(latitudes, longitudes, temperature_data, save_filename=None,
hour=None, target_lat=None, target_lon=None):
    temperature_celsius = temperature_data - 273.15

    if target_lat is not None and target_lon is not None:
        ilat = np.argmax(latitudes > target_lat)
        ilon = np.argmax(longitudes > target_lon)

        lat_extent = latitudes[ilat - 5: ilat + 5]
        lon_extent = longitudes[ilon - 5: ilon + 5]

        plt.contourf(lon_extent, lat_extent, temperature_celsius[ilat - 5: ilat + 5,
ilon - 5: ilon + 5], cmap='viridis')
    else:
        plt.contourf(longitudes, latitudes, temperature_celsius, cmap='viridis')

    plt.colorbar()

    title = 'Temperature at 2m (°C)'

    if target_lat is not None and target_lon is not None:
        title += f' - Around {target_lat}, {target_lon}'

    if hour:
        title += f' - Hour {hour}'

    plt.title(title)
    plt.xlabel('Longitude')
    plt.ylabel('Latitude')

    if save_filename:
        plt.savefig(save_filename)
        plt.close()
```

Η συνάρτηση get_variable.

```
def get_variable(file_path, variable_name, latitude, longitude):
    hour_list = []
    variable_list = []

    sorted_files = sorted(os.listdir(file_path))
    for file in sorted_files:
        if file.endswith(".grib2"):
            hour_match = re.search(r'_(\d{3})_', file)
            if hour_match:
                hour = hour_match.group(1)
                gribfile = os.path.join(file_path, file)

                try:
                    ds = cfgrib.open_dataset(gribfile)
                except cfgrib.errors.CfError:
                    continue

                if variable_name in ds.variables:
                    variable_data = np.array(ds[variable_name])
                    latitudes = np.array(ds['latitude'])
                    longitudes = np.array(ds['longitude'])

                    interp = RegularGridInterpolator((latitudes, longitudes),
variable_data)
                    variable = interp([latitude, longitude])[0]

                    hour_list.append(hour)
                    variable_list.append(variable)

    return hour_list, variable_list
```

Η συνάρτηση process_weather_data.

```
def process_weather_data(location_lat, location_lon, files_path, variable_name,
output_message):
    temperature_data_list = []
    hour_list = []
    temperature_list = []

    sorted_files = sorted(os.listdir(files_path))
    for file in sorted_files:
        if file.endswith(".grib2"):
            hour_match = re.search(r'_(\d{3})_', file)
            if hour_match:
                hour = hour_match.group(1)
                gribfile = os.path.join(files_path, file)

                original_stderr = sys.stderr
                sys.stderr = open(os.devnull, 'w')  # Απενεργοποίηση εμφάνισης μηνυμάτων
σφάλματος για αδύνατα αρχεία
                try:
                    ds = cfgrib.open_dataset(gribfile)
                except cfgrib.errors.CfError:
                    continue
                finally:
                    sys.stderr = original_stderr

                if variable_name in ds.variables:
                    temperature_data = np.array(ds[variable_name])
                    latitudes = np.array(ds['latitude'])
                    longitudes = np.array(ds['longitude'])

                    ilat = np.argmax(latitudes > location_lat)
                    ilon = np.argmax(longitudes > location_lon)

                    temperature_celsius = temperature_data[ilat - 1, ilon - 1] - 273.15

                    print(f"{output_message} at {location_lat}, {location_lon} at {hour}
Hours is {temperature_celsius:.2f}°C.")

                    temperature_data_list.append(temperature_data)
                    hour_list.append(int(hour))
                    temperature_list.append(temperature_celsius)

                    save_filename = os.path.join(files_path,
f"visualization_{hour}.png")
                    visualize_temperature(latitudes, longitudes, temperature_data,
save_filename, hour=hour, target_lat=location_lat, target_lon=location_lon)

                else:
                    print(f"{output_message} variable not found in the dataset for
{file}. Please check the variable names.")

                ds.close()

    if temperature_data_list:
        latitudes = np.array(ds['latitude'])
        longitudes = np.array(ds['longitude'])
        all_temperature_data = np.mean(temperature_data_list, axis=0)

        plt.figure()
```

```
        visualize_temperature(latitudes, longitudes, all_temperature_data,
target_lat=location_lat, target_lon=location_lon)
        plt.title(f'Temperature at {location_lat}, {location_lon} over Time')
        plt.xlabel('Longitude')
        plt.ylabel('Latitude')
        plt.show()

        plt.figure()
        plt.plot(hour_list, temperature_list, marker='o')
        plt.title(f'Temperature at {location_lat}, {location_lon} over Time')
        plt.xlabel('Hour')
        plt.ylabel('Temperature (°C)')
        plt.grid(True)
        plt.show()
```

Η συνάρτηση create_video.

```
def create_video(files_path, video_output_path):
    images = []
    sorted_files = sorted(os.listdir(files_path))
    for file in sorted_files:
        if file.startswith("visualization") and file.endswith(".png"):
            image_path = os.path.join(files_path, file)
            images.append(image_path)

    if not images:
        print("No PNG files found for creating the video.")
        return

    height, width, layers = cv2.imread(images[0]).shape
    video_output_path = os.path.splitext(video_output_path)[0]
    video_output_path += ".mp4"
    video = cv2.VideoWriter(video_output_path, cv2.VideoWriter_fourcc(*'mp4v'), 1,
(width, height))

    for image in images:
        img = cv2.imread(image)
        video.write(img)

    cv2.destroyAllWindows()
    video.release()
    print(f"Video created and saved: {video_output_path}")
```

Χρησιμοποιούμενες βιβλιοθήκες.

```
import os
import requests
from bs4 import BeautifulSoup
import bz2
import cfgrib
import numpy as np
import platform
import re
import cv2
from matplotlib import pyplot as plt
```

Δημιουργία γραφημάτων-βίντεο

```
files_path_1 = os.path.join(base_directory,
os.path.basename(downloaded_directory_urls[0].rstrip('/')))
    print_larissa_temperature(files_path_1, "t2m", "Temperature")

    video_output_path = os.path.join(base_directory, "larissa_temperature_video")
    create_video(files_path_1, video_output_path, first_file_date)
```