ΔΙΕΘΝΕΣ ΠΑΝΕΠΙΣΤΗΜΙΟ ΤΗΣ ΕΛΛΑΔΟΣ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

«Ανάπτυξη πρόσθετου του WordPress για την διαχείριση προγράμματος μαθημάτων, εξετάσεων, εκδηλώσεων»

Της φοιτήτριας
Κρίκου Μαριάννα
Αρ. Μητρώου: 113741

Επιβλέπων Καθηγητής
Ονοματεπώνυμο: Ευστάθιος Αντωνίου

Ημερομηνία 20/01/2021

Τίτλος Π.Ε.: Ανάπτυξη πρόσθετου του WordPress για την διαχείριση προγράμματος μαθημάτων, εξετάσεων, εκδηλώσεων
Κωδικός Π.Ε. 19077
Ονοματεπώνυμο φοιτητή: Μαριάννα Κρίκου
Ονοματεπώνυμο εισηγητή Ευστάθιος Αντωνίου
Ημερομηνία ανάληψης Δ.Ε 09/12/2019
Ημερομηνία περάτωσης Δ.Ε 20/01/2021

Βεβαιώνω ότι είμαι ο συγγραφέας αυτής της εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης, έχω καταγράψει τις όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών, εικόνων και κειμένου, είτε αυτές αναφέρονται ακριβώς είτε παραφρασμένες. Επιπλέον, βεβαιώνω ότι αυτή η εργασία προετοιμάστηκε από εμένα προσωπικά, ειδικά ως διπλωματική εργασία, στο Τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του ΔΙ.ΠΑ.Ε.

Η έγκριση της πτυχιακής εργασίας από το Τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του Διεθνούς Πανεπιστημίου της Ελλάδος, δεν υποδηλώνει απαραιτήτως και αποδοχή των απόψεων του συγγραφέα, εκ μέρους του Τμήματος.

ΔΙΕΘΝΕΣ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΤΗΣ ΕΛΛΑΔΟΣ

ΑΛΕΞΑΝΔΡΕΙΑ ΠΑΝΕΠΙΣΤΗΜΙΟΥΠΟΛΗ
ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ
Δ/νση: ΤΘ: 141, 57400 Σίνδος, Θεσσαλονίκη, Τηλ: 2310 013621, Fax: 2310 791132

Προς: Γραμματεία του Τμήματος Μηχανικών Πληροφορικής & Ηλεκτρονικών Συστημάτων

ΒΕΒΑΙΩΣΗ ΕΞΕΤΑΣΗΣ ΔΙΠΛΩΜΑΤΙΚΗΣ ΕΡΓΑΣΙΑΣ

Η Κρίκου Μαριάννα με όνομα πατρός Κρίκος Δημήτριος φοιτήτρια του τμήματος Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του ΔΙ.ΠΑ.Ε. με Α.Μ. 113741 εξετάστηκε σήμερα στη διπλωματική εργασία με θέμα Ανάπτυξη πρόσθετου του WordPress για την διαχείριση προγράμματος μαθημάτων, εξετάσεων, (κωδικός Π.Ε. 19077) και επιβλέποντα τον/την κ. Ευστάθιο Αντωνίου ενώπιον τριμελούς εξεταστικής επιτροπής και αξιολογήθηκε ως:

ΒΑΘΜΟΛΟΓΙΑ

Α/Α	*Εξεταστής*	*Ανάπτυξη/Μεθοδολογία.* Βαθμός (0-10) x 0,3	*Εκπλήρωση στόχων.* Βαθμός (0-10) x 0,3	*Ποιότητα γραπτού κειμένου.* Βαθμός (0-10) x 0,2	*Παρουσίαση.* Βαθμός (0-10) x 0,2	*Ολικός βαθμός*	*Υπογραφή*
1							
2							
3							

Γενικός βαθμός της πτυχιακής εργασίας_____(_____)

Ο επιβλέπων της Π.Ε.

_____

(Ημερομηνία, υπογραφή)

Περίληψη

Το παρόν κείμενο αναφέρεται στο θέμα «Ανάπτυξη πρόσθετου του WordPress για την διαχείριση προγράμματος μαθημάτων, εξετάσεων, εκδηλώσεων», δηλαδή τη δημιουργία ενός plugin στο περιβάλλον του WordPress. Στις επόμενες σελίδες υπάρχουν αρχικά κάποιες γενικές πληροφορίες για την ανάπτυξη ιστοσελίδας και τη χρήση των content management systems, με μεγάλη έμφαση στο WordPress. Έπειτα ακολουθεί ένας σύντομος οδηγός για τη αρχή της δημιουργίας ενός πρόσθετου στο συγκεκριμένο περιβάλλον. Γίνεται μια μικρή αναφορά στις βιβλιοθήκες που έχουν χρησιμοποιηθεί αλλά μετά από αυτό το υπόλοιπο κομμάτι είναι για το ίδιο το πρόσθετο.
Πρώτα υπάρχει ένας οδηγός για το χρήστη του WordPress, για το πως θα εγκαταστήσει το πρόσθετο και πως θα το χρησιμοποιήσει, κάτι το όποιο υπάρχει και μέσα στο ίδιο το πρόσθετο.
Μετά ακολουθεί ένα κομμάτι που απασχολεί άτομα που θέλουν να μάθουν για το πως λειτουργεί ο κώδικας πίσω από όλο αυτό.
Και τέλος έχω προσθέσει ένα κεφάλαιο με τη πορεία της εργασίας , τις δυσκολίες που αντιμετώπισα κατά τη διάρκεια αυτής και κάποιες μελλοντικές επεκτάσεις που θα μπορούσαν να γίνουν.

Abstract

This text refers to the topic "Development of a WordPress plugin for managing courses, exams, events", i.e. creating a plugin in the WordPress environment. In the following pages there is initially some general information about web development and the use of content management systems, with great emphasis on WordPress. The following is a brief guide to the beginning of creating an add-on in that environment. A little reference is made to the libraries that have been used but after this the rest is for the plugin itself.
First there is a guide for the WordPress user, on how to install the plugin and how to use it, something that exists in the plugin itself.
This is followed by a section that if for people who want to know how the code works behind it all.
And last but not least I have added a chapter on the course of the work, the difficulties I encountered during it and some future extensions that could be made.

Περιεχόμενα

ΒΕΒΑΙΩΣΗ ΕΞΕΤΑΣΗΣ ΔΙΠΛΩΜΑΤΙΚΗΣ ΕΡΓΑΣΙΑΣ 3
Περίληψη 4
Abstract 5
Εισαγωγή 4
1. Ανάπτυξη Ιστοσελίδας 5
1.1. Γλώσσες 5
1.2. CMS 7
1.3. WordPress 7
2. Δημιουργία Πρόσθετου 10
2.1. Hooks 11
2.2. WPDB 12
3. Βιβλιοθήκες και Εργαλεία 16
3.1. FullCalendar 16
3.2. Datatables 16
3.3. iCalendar 17
3.4. VS Code 17
3.5. Git Hub 18
4. Εφαρμογή 19
4.1. Λειτουργία 19
4.2. Εγκατάσταση σε WordPress 19
4.3. Χρήση του Plugin 20
4.4. Δομή 23
4.5. Στη Πράξη 27
5. Συζήτηση 34
5.1. Πορεία 34
5.2. Δυσκολίες 35
5.3. Μελλοντικές Επεκτάσεις 36
Βιβλιογραφία και Αναφορές 37
Παράρτημα 38

Περιεχόμενα Εικόνων

Εικόνα 1 - Εγκατάσταση WordPress 10
Εικόνα 2- Παράδειγμα Hook 12
Εικόνα 3 - Πίνακες WordPress 15
Εικόνα 4 - Αρχεία Full Calendar 16
Εικόνα 5 - Αναφορά σε Datatables library 16
Εικόνα 6 – MyCal στο Dashboard 19
Εικόνα 7 - Εγκατάσταση σε server 19
Εικόνα 8 – Το πρόσθετο στο WordPress 20
Εικόνα 9 – Εμφάνιση Events 20
Εικόνα 10 - Δομή Πρόσθετου 23
Εικόνα 11 - Ανανέωση JSON 25
Εικόνα 12 - Μέθοδος για αντιγραφή 27
Εικόνα 13 - Δημιουργία πινάκων 27
Εικόνα 14 - Δημιουργία Page 28
Εικόνα 15 - Δημιουργία Menu 28
Εικόνα 16 - Έλεγχος ημερομηνίας και ώρας 29
Εικόνα 17 - Έλεγχος ίδιου τίτλου και χρώματος στην εισαγωγή κατηγορίας 29
Εικόνα 18 - Μέθοδος για εισαγωγή κατηγορίας 29
Εικόνα 19 - Pop-up για επεξεργασία event 30
Εικόνα 20 - Προβολή ενός event στο πίνακα 31
Εικόνα 21 - Μέθοδος για ανανέωση event πριν το σβήσιμο μιας κατηγορίας 31
Εικόνα 22 - Μέθοδος δημιουργίας JSON για την εμφάνιση κατηγορίας στο Pop-up επεξεργασίας 31
Εικόνα 23 - Ορισμός των Shortcodes 32
Εικόνα 24 - Shortcode attributes 32
Εικόνα 25 - Χρήση των Attributes 33
Εικόνα 26 - For loop για τη δημιουργία ενός αρχείου με περιεχόμενο σε JSON μορφή των κατηγοριών που έχουν δοθεί στο shortcode 33
Εικόνα 27 - Παλιό πλάνο για τους πίνακες του πρόσθετου 34

Εισαγωγή

Όταν πήρα αυτό το θέμα πτυχιακής δεν είχα σχεδόν καθόλου εμπειρία με PHP εκτός του ότι είχα κάνει στη σχολή πόσο μάλλον για τη δημιουργία πρόσθετου. Το πρόσθετο αρχικά ήταν λίγο στοχευμένο για την ιστοσελίδα της σχολής αλλά στη τελική το έφτιαξα με τέτοιο τρόπο ώστε να είναι κάτι πιο γενικό και να μπορεί να ενταχθεί σε οποιαδήποτε σελίδα . Ασχολείται με τη δημιουργία προγράμματος για την οργάνωση εκδηλώσεων οποιουδήποτε τύπου είτε μαθήματα είτε εξετάσεις ή οτιδήποτε μπορεί να μπει σε ένα ημερολόγιο. Στη διάρκεια της εργασίας αναφέρομαι στην εισαγωγή events και όχι εκδηλώσεις καθώς η λέξη events είναι κάτι πιο γενικό και έχει και άμεση σχέση με το πρόσθετο καθώς έτσι ονομάζω και το πίνακα στη βάση. Μέσα σε αυτό το χρόνο που έχω το συγκεκριμένο θέμα πτυχιακής έμαθα αρκετά γύρω από τη PHP και το πως επικοινωνεί με το WordPress. Πιστεύω ότι η δημιουργία προσθέτου είναι αρκετά ενδιαφέρον . Υπάρχουν άπειρα πρόσθετα στο WordPress όπως και αρκετά που προσφέρουν παρόμοια πράγματα με το δικό μου, αν και όσο έψαξα το τι υπάρχει δεν βρήκα κανένα που να είναι δωρεάν ή έστω με premium εκδοχή. Στη τελική στο πρόσθετο κάνει σωστά τις λειτουργίες που χρειάζεται και έμαθα αρκετά για PHP αλλά και για JavaScript.

1. Ανάπτυξη Ιστοσελίδας

Στις μέρες το ίντερνετ είναι για τη πλειοψηφία το βασικό μέσο ενημέρωσης, αγορών ακόμα και επαφή με άλλα άτομα. Από ένα blog μέχρι το μαγαζί που θα παραγγείλεις για να φας υπάρχουν άπειρες ιστοσελίδες στο διαδίκτυο.
Όλο και περισσότερες διεργασίες που κάποτε γίνονταν από κοντά πλέον μπορούν να γίνουν από τον υπολογιστή. Και τη περίοδο της πανδημίας αυτό έχει γίνει ακόμα πιο σημαντικό. Κάθε εμπορικό μαγαζί, εστίασης, φροντιστήριο και οποιαδήποτε άλλη επιχείρηση θέλει να έχει μια διαδικτυακή παρουσία. Η ανάγκη για δημιουργία σελίδων δεν σταματάει ποτέ αντίθετα αυξάνεται.
Το web development είναι ένας από τους διαδεδομένους κλάδους της πληροφορικής, όπου web εννοούμε Word Wide Web ή WWW. Αναφέρεται στην δημιουργία αλλά και την συντήρηση ιστοσελίδων. Υπάρχουν διαφορετικοί τρόποι για την δημιουργία και την ανάπτυξη μιας ιστοσελίδες και διάφορες γλώσσες που τα υποστηρίζουν.

Ο όρος “web developer” και “web designer” χρησιμοποιούνται συχνά συνώνυμα, δεν είναι όμως καθώς ο ρόλος ενός web designer είναι να σχεδιάσει την ιστοσελίδα και ο developer να την χτίσει η μάλλον ήταν. Πλέον οι εποχές άλλαξαν και υπάρχει η δυνατότητα σχεδιασμού και δημιουργίας μέσω open source εργαλείων.

Η ανάπτυξη μιας ιστοσελίδας περιλαμβάνει πολλούς τρόπους δημιουργίας περιεχομένου. Μπορεί μια ιστοσελίδα να γραφτεί στο χέρι σε κώδικα σε κάποιον κειμενογράφο η σε προγράμματα όπως Dreamweaver η μέσω open source εργαλείων. Στην κορυφή βρίσκεται το WordPress με περισσότερες από 27 εκατομμύρια ενεργές εγκαταστάσεις που αποτελούν το 34% των ιστοσελίδων και το 62% μεταξύ των CMS μετέπειτα ακολουθεί το Wix με 3.8 εκατομμύρια ενεργές εγκαταστάσεις τρίτο το Squarespace με 2.2 εκατομμύρια 4ο το Joomla 1.5 εκατομμύριά.

1. ***Front-End***

   Front end λέμε το μέρος της ιστοσελίδας οπού ο χρήστης έχει την επικοινωνία η οτιδήποτε βλέπει όταν είναι στο internet από γραμματοσειρές, χρώματα, dropdown menu και sliders ένας όμορφος συνδυασμός από HTML που αποτελεί σε εισαγωγικά τον σκελετό της ιστοσελίδας, την CSS που είναι η σάρκα και JavaScript που του δίνει πνοή.

2. ***Back-End***

   Όταν μιλάμε για back end μιλάμε για μαύρη μαγεία και από πίσω κρύβονται οι MySQL , Oracle και SQL, και εννοούμε την δημιουργία ενός server ή και μιας βάσης δεδομένων που επικοινωνούν μεταξύ τους μέσω PHP, Ruby, Python, Java και .Net ή διάφορα PHP frameworks.

1.1. Γλώσσες

Οι γλώσσες χωρίζονται σε δύο κατηγορίες, σε γλώσσες προγραμματισμού και σε γλώσσες σήμανσης. Οι γλώσσες προγραμματισμού αποτελούνται από αλγόριθμους και εντολές που δίνουν οδηγίες στον υπολογιστή για το τι να κάνει, ενώ οι γλώσσες σήμανσης δεν περιέχουν αλγοριθμική. Οι γλώσσες σήμανσης είτε θα εμφανίσουν κάτι, όπως η HTML είτε θα περιέχουν κάποια δεδομένα για εμφάνιση όπως η XML. Υπάρχει όμως και μια τρίτη κατηγορία ή ακόμα καλύτερα μια υπό κατηγορία στις γλώσσες προγραμματισμού, οι scripting γλώσσες ή γλώσσες σεναρίου. Τέτοιες γλώσσες λέμε την PHP, την Python , την JavaScript και άλλες. Οι script γλώσσες κατά βάση συνυπάρχουν με άλλες γλώσσες και συνδέουν μια γλώσσα με μια άλλη.

1. ***HTML***

   Η HTML (Hyper Text Markup Language) είναι μια γλώσσα σήμανσης και είναι το πρώτη επαφή που έχουν οι περισσότεροι για τη δημιουργία ιστοσελίδας. Όπως ήδη αναφέρθηκε η HTLM δίνει το σκελετό μιας ιστοσελίδας. Δημιουργήθηκε από τον Tim Berners-Lee στο ερευνητικό ινστιτούτο CERN στην Ελβετία. Από το 2014 έχει δημιουργηθεί η HTML5 σαν επέκταση της κλασσικής HTML και προσδέθηκαν λειτουργίες όπως διαφορά multimedia tags (<video>, <audio>). Παρόλο που η HTML5 είναι και αυτή μια γλώσσα σήμανσης περιέχει κάποια χαρακτηριστικά που θυμίζουν γλώσσα προγραμματισμού.

   Στο plugin η HTML χρησιμοποιείτε στην δημιουργία των τεσσάρων σελίδων που εμφανίζονται στο Dashboard του WordPress.

2. ***JavaScript***

   Η JavaScript είναι γλώσσα προγραμματισμού και παρόλο που έχει κάποια κοινά στο κώδικα και στο όνομα δεν έχει σχέση με τη Java. Η JavaScript μπορεί είτε να γραφτεί μέσα σε ένα html αρχείο μέσα σε <script> </script> είτε να έχει δικό της αρχείο με κατάληξη js και το αρχείο html να το διαβάσει.

   Η JavaScript είναι μια interpreted γλώσσα. Αυτό σημαίνει ότι ο κώδικας μπορεί να εκτελεστεί γραμμή-γραμμή, και δεν χρειάζεται να γίνει compile όλο το αρχείο.

   Αξίζει να αναφερθεί ότι ένα πολύ σημαντικό κομμάτι της JavaScript είναι η ajax. Η ajax (Asynchronous JavaScript And XML) δεν είναι ακόμη μια γλώσσα αλλά είναι μια σειρά από μεθοδολογίες της JavaScript. Ένα από τα πιο σημαντικά features που προσφέρουν αυτές οι μεθοδολογίες είναι η ενημέρωση κάποιου στοιχείου της σελίδας χωρίς να χρειάζεται να ανανεώσουμε ολόκληρη τη σελίδα. Αν και το ίδιο το όνομα αναφέρεται σε XML η ajax έχει πιο συχνά επαφή με αρχεία JSON.

   Εκτός από τις βιβλιοθήκες JavaScript που χρησιμοποιεί το plugin οι οποίες θα αναλυθούν αργότερα, χρησιμοποιείτε στη σελίδα Shortcodes για να αλλάζουν δυναμικά τα πεδία που αλλάζει ο χρήστης.

3. ***CSS***

   Η CSS (Cascading Style Sheets) όπως JavaScript μπορεί να είναι είτε σε στο ίδιο αρχείο ανάμεσα σε <style> </style> ή σε ξεχωριστό αρχείο με κατάληξη css. Η CSS χρησιμοποιείτε για τη μορφοποίηση των σελίδων μέγεθος, χρώμα ,κενά(margins) και άλλα. Η CSS σε συνδυασμό με τη JavaScript δίνουν δυναμικότητα σε μια ιστοσελίδα και περισσότερη διάδραση με το χρήστη, όπως για παράδειγμα να αλλάζει το χρώμα ενός κουμπιού όταν ο χρήστης περάσει το ποντίκι από αυτό.

   Στο plugin χρησιμοποιείται στις βασικές σελίδες που βλέπει ο χρήστης.

4. ***PHP***

   Η PHP (Hypertext Preprocessor) είναι μια διαδικτυακή Server-Side γλώσσα προγραμματισμού. Σε αντίθεσή με HTML, JavaScript και CSS που το μόνο που χρειάζεται για να κάνεις δοκιμές τοπικά είναι ένα Text Editor, για να τρέξεις ένα script με PHP θα χρειαστεί ένα local server environment όπως το XAMPP και το WAMPP. Η php μπορεί να είναι και αυτή μέσα σε μία HTML σελίδα ανάμεσα σε <?php ?> ή και τελείως μόνη της όπως για παράδειγμα να τραβάει με τη μέθοδο POST ότι στέλνει η HTML από μία φόρμα. Οι μεταβλητές ξεκινούν με το σύμβολο του δολαρίου($) και όπως και στη JavaScript δεν χρειάζεται να ορίσουμε το τύπο μιας μεταβλητής.

   Το plugin είναι κατά βάση γραμμένο σε PHP αφού και το WordPress είναι.

5. ***MySql***

   Η MySql είναι ένα Relational Database Management System ή εν συντομία RDBMS που χρησιμοποιεί SQL. Η SQL είναι γλώσσα για τη διαχείριση βάσεων δεδομένων. Η MySql συνδέεται ευκολά με την PHP η οποία μπορεί να "τρέξει" SQL Statements όπως πχ "SELECT * FROM table" για να τραβήξουμε όλα τα στοιχεία από ένα πίνακα στη βάση με όνομα table.

6. JSON

Η JSON (JavaScript Object Notation) μια μορφή αρχείου για τη αποθήκευση δεδομένων σε πολύ μικρό μέγεθος. Είναι πολύ εύκολή στην ανάγνωση, μπορούν να δεδομένα να είναι γραμμένα είτε σε μορφή πίνακα, διανύσματος, λίστας ή και ακολουθίας.

Στο plugin δημιουργούνται κάποια JSON αρχεία για την εμφάνιση event στο fullcalendar αλλά και για την επεξεργασία καταχωρήσεων.

1.2. CMS

1. Τι είναι

Ένα CMS χρησιμοποιείται για να δημιουργείς ιστοσελίδες χωρίς κώδικα η να δημοσιεύεις άρθρα γρήγορα.

Παλιότερα προκειμένου να δημιουργήσεις μια ιστοσελίδα έπρεπε να γνωρίζεις HTML πράγμα που ήταν δύσκολο για το περισσότερο κόσμο, για αυτό και ήρθαν στα χέρια μας τα CMS, για να μας κερδίσουν χρόνο και να δώσουν την δυνατότητα σε ακόμα περισσότερο κόσμο να δημιουργήσει αυτό που θέλει.

Το CMS είναι ένα ισχυρό εργαλείο που σου επιτρέπει να δημιουργήσεις ένα Blog, μια ιστοσελίδα, ένα E-commerce κατάστημα, το προσωπικό σου portfolio και πολλά ακόμα.

Κάποια από τα πιο διαδεδομένα CMS είναι:

- WordPress
- Joomla
- Drupal
- Magento
- Squarespace
- Wix
- TYPO3

2. Content Editor

Τα CMS περιέχουν συνήθως έναν Content editor για να δημιουργούν το υλικό τους.

Συνήθως επικρατεί ο κειμενογράφος WYSIWYG που σημαίνει What you see is what you get κατα συνέπεια αυτό που βλέπεις είναι και το αποτέλεσμα που θα υπάρχει στο frontend και παραπέμπει ιδιαίτερα σε Word με δυνατότητες αποθήκευσης και χρήσης φωτογραφιών.

1.3. WordPress

Το WordPress θεωρείται το πιο απλό και το πιο δημοφιλή εργαλείο για να δημιουργείς ιστοσελίδες η blog. Για την ακρίβεια το WordPress κατέχει το 39.5% των ιστοσελίδων στο διαδίκτυο.
Σε πιο τεχνικό επίπεδο το WordPress είναι ένα open source content management system licensed under GPLv2 που σημαίνει ότι ο καθένας μπορεί να το τροποποιήσεις δωρεάν. Ένα Content Management System ουσιαστικά είναι ένα εργαλείο που κάνει πιο εύκολη την οργάνωση της ιστοσελίδας και του περιεχομένου χωρίς να χρειάζεται να ξέρεις τίποτα από προγραμματισμό.

Πολλά χρόνια πριν το WordPress ήταν ένα εργαλείο για να δημιουργείς blog και όχι ιστοσελίδες. Αλλά αυτό άλλαξε με τον καιρό καθώς πλέον έχει αλλάξει τόσο πολύ ο κώδικας του που σχεδόν τα πάντα είναι εφικτά.

Ο δημοφιλέστερος τρόπος για να δημιουργήσεις

- Επαγγελματική ιστοσελίδα
- eCommerce κατάστημα
- Blog
- Portfolio
- Βιογραφικό
- Forum
- Κοινωνικά δίκτυα
- Site μελών

Και πολλά ακόμα.

1. Πλεονεκτήματα

- Τιμή
  Το WordPress από μόνο του είναι ένα δωρεάν CMS και ένα μεγάλο ποσοστό των θεμάτων και των προσθέτων είναι και αυτά δωρεάν ή έχουν πολύ χαμηλή τιμή.

- Φιλικό στον χρήστη.
  Καθώς και δεν είμαστε όλοι στο ίδιο επίπεδο και στην ίδια άνεση σχετικά με τον κώδικα τα CMS επιτρέπουν σε όλους να έχουν μια καλύτερη ευκαιρία για να εμφανιστούν στον κόσμο αλλά και αυτοί που είναι γνώστες μπορούν να ωφεληθούν γλιτώνοντας χρόνο. Το WordPress έχει ένα πολύ εύκολο dashboard το οποίο δεν απαιτεί καθόλου γνώσεις πάνω σε προγραμματισμό για τη διαχείριση του. Αλλά κάποιος με γνώσεις πάνω σε προγραμματισμό μπορεί να κάνει ακόμα περισσότερα πράγματα.

- Συχνές αναβαθμίσεις
  Το WordPress είναι ένα εργαλεία που συνεχώς εξελίσσεται με αποτέλεσμα να υπάρχει καινούρια έκδοση πολύ συχνά και νέες λειτουργίες και δυνατότητες Οι αναβαθμίσεις μπορούν να γίνουν αυτόματα σε κάποιες περιπτώσεις

- Πολλαπλοί χρήστες
  Το Dashboard του WordPress δίνει το δυνατότατά σε πολλούς χρήστες μπορούν να είναι ταυτόχρονα σε μια σελίδα και να κάνουν αλλαγές η να δημιουργούν υλικό, ο μόνος περιορισμός είναι να μην βρίσκονται στο ίδιο page του WordPress.

- Φιλικό στις μηχανές αναζήτησης
  Το να προβληθείς στο ίντερνετ μπορεί να μην σημαίνει απλά να έχεις μια βιτρίνα αλλά να εμφανίζεσαι και στις μηχανές αναζήτησης.

- Υλικό
  Το WordPress χρησιμοποιείται από πάρα πολύ κόσμο με αποτέλεσμα να υπάρχει πολύ υλικό στο διαδίκτυο για το πως χρησιμοποιείται και λύσεις για διάφορα προβλήματα. Υπάρχουν forums που ασχολούνται μόνο με θέματα το WordPress ακόμα και πολλά εκπαιδευτικά video στο YouTube. Είναι δηλαδή πολύ εύκολο για κάποιος που τώρα ξεκινάει να μαθαίνει να βρει υλικό για να τον βοηθήσει

2. Μειονεκτήματα

- Μπορεί να κοστίζει
  Κάποια θέματα και πρόσθετα που προσφέρουν πολλά παραπάνω πράγματα κοστίζουν.

- Ασφάλεια
  Ίσως το μεγαλύτερο μειονέκτημα ενός CMS είναι η ασφάλεια του. Περισσότερο από το 1/3 όλων των ιστοσελίδων είναι σε open source CMS, 75% των επιθέσεων έγινε σε wordpress ιστοσελίδες, ο ανοιχτός κώδικας είναι αυτό που κάνει μια σελίδα πιο λαχταριστή και πιο ευάλωτη στους χάκερς καθώς και τους δίνεται η δυνατότητα να μελετήσουν την ασφάλεια μιας σελίδας πριν κάνουν οτιδήποτε.

- Συνεχής επίβλεψη
  Πολλές φορές είναι πιο αποτελεσματικό να έχεις κάποιον να διαχειρίζεται τις αναβαθμίσεις, απαντήσεις σε μηνύματα κτλ. Ίσως οι πιο σημαντικοί παράγοντες για να μην παραμελούνε καθώς το μεγαλύτερο ποσοστό ιστοσελίδων που χακαραν δεν ήταν αναβαθμισμένες στην τελευταία έκδοση. Αλλά με οποιοδήποτε τρόπο να είναι φτιαγμένο ένα site πάντα θα πρέπει να υπάρχει και κάποιος που το επιβλέπει.

3. Θέματα

Η εμφάνιση μιας ιστοσελίδας είναι από τα πρώτα πράγματα που θα προσέξει κάποιος, είναι σαν το εξώφυλλο ενός βιβλίου. Το WordPress χρησιμοποιεί θέματα (themes) τα οποία είναι φτιαγμένα σε CSS και προσφέρουν πολύ περισσότερες επιλογές στην εμφάνιση αλλά και στη διαχείριση της σελίδας. Υπάρχουν πάρα πολλά θέματα διαθέσιμα στο χρήστη, κάποια δωρεάν, για διάφορους τύπους ιστοσελίδας. Ένα WordPress θέμα μπορεί αλλάξει τα χρώματα, την διάταξη, τα μενού ακόμα και το τρόπου που μπορεί να επεξεργαστεί ο χρήστης τα pages στο dashboard.

Όταν ένας χρήστης θέλει να κάνει αλλαγές σε ένα θέμα που έχει κατεβάσει συνιστάτε πρώτα να αντιγράψει στο φάκελο που θέματος και μετά να κάνει ότι αλλαγές θέλει στον καινούριο φάκελο έτσι ώστε αν κάτι πάει λάθος να μπορεί να ξεκινήσει από την αρχή.

4. Πρόσθετα

Τα θέματα δίνουν στη σελίδα μια καλύτερα εμφάνιση και τα πρόσθετα δίνουν παραπάνω λειτουργίες. Υπάρχει ένα ρητό που λέει «Υπάρχει ένα πρόσθετο για αυτό», δηλαδή ότι για σχεδόν ότι λειτουργία θέλει ο χρήστης να προσθέσει στη σελίδα του θα υπάρχει κάποιο πρόσθετο που να μπορεί να ανταπεξέλθει. Υπάρχουν πάρα πολλά πρόσθετα και πολλά από αυτά είναι και δωρεάν αλλά όσο πιο πολλές και περίπλοκες λειτουργίες προσφέρει τόσο πιο ακριβό μπορεί να είναι. Πολλά και από τα δωρεάν αλλά προσφέρουν περισσότερες λειτουργίες με τη premium εκδοχή η οποία θα είναι με πληρωμή.

2. Δημιουργία Πρόσθετου

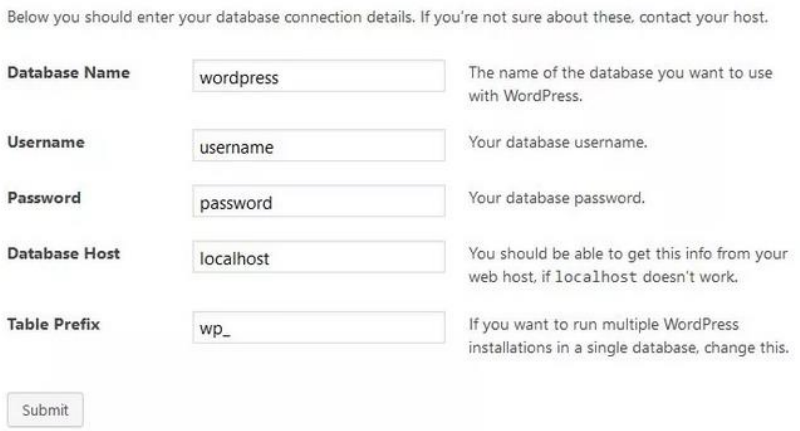
Για τη δημιουργία ενός πρόσθετου απαιτούνται αρχικά βασικές γνώσεις PHP αφού το ίδιο το WordPress είναι γραμμένο σε PHP. Έπειτα για την διαχείριση του προσθετού από τον χρήστη θα χρειαστεί HTML και ίσως CSS και JavaScript.

Πριν την δημιουργία του πρόσθετου θα χρειαστεί ένα περιβάλλον WordPress. Αυτό μπορεί να γίνει είτε τοπικά είτε σε κάποιο σέρβερ. Άλλα για να μπορούμε να κάνουμε εύκολα αλλαγές και να βλέπουμε απευθείας το αποτέλεσμα αυτών των αλλαγών είναι προτιμότερο να δουλεύουμε τοπικά. Αφού εγκαταστήσουμε ένα local server environment (XAMMP ή WAMPP) θα χρειαστεί να κατεβάσουμε το WordPress και να το τοποθετήσουμε το φάκελο htdocs που έχει δημηγορεί στο δίσκο μετά την εγκατάσταση.

Επόμενο βήμα είναι η εγκατάσταση του WordPress. Αφού ανοίξουμε το το XAMPP ή το WAMPP ενεργοποιούμε τις επιλογές Apache και MySql, πλέον έχουμε πρόσβαση στο σύνδεσμο του phpMyAdmin, συνήθως θα είναι στο localhost:8080/phpmyadmin/ αλλά μπορεί και να διαφέρει.

Μέσα στο phpMyAdmin θα πρέπει να δημιουργήσουμε μια βάση με ότι όνομα θέλουμε, και μετά με τον ίδιο τρόπο που πήγαμε στο phpMyAdmin μπορούμε να πάμε και στο WordPress (localhost:8080/wordpress/ αν δεν αλλάξαμε το όνομα του φακέλου στο htdocs) και εκεί θα ξεκινήσει η εγκατάσταση.

Η εγκατάσταση του WordPress είναι αρκετά απλή, θα ζητηθεί από το χρήστη να δώσει το όνομα της βάσης που μόλις έφτιαξε, το όνομα διαχειριστή της βάσης και το κωδικό, το Host και ένα Prefix για τους πίνακες που θα δημιουργηθούν.
Τέλος θα ζητήσει από το χρήστη να φτιάξει ένα λογαριασμό για να συνδέετε. Μετά από αυτό το WordPress είναι έτοιμο για πειράματα.



Εικόνα 1 - Εγκατάσταση WordPress

Για να ξεκινήσουμε το πρόσθετο θα φτιάξουμε πρώτα ένα φάκελο μέσα στο φάκελο του WordPress, συγκεκριμένα στο φάκελο Plugins (wordpress\wp-content\plugins) με το όνομα που θέλουμε και δημιουργούμε μέσα ένα αρχείο PHP τίτλο ίδιο με του φακέλου. Για να μπορεί το WordPress να αναγνωρίσει το φάκελο αυτό ως πρόσθετο θα πρέπει να υπάρχει μέσα ένα σχολείο με τα εξής

```
/**
* Plugin Name:       My Basics Plugin
* Description:       Handle the basics with this plugin.
* Version:           1.10.3
* Author:            John Smith
* Author URI:        https://author.example.com/
*/
```

Πλέον το WordPress αναγνωρίζει το αρχείο ως πρόσθετο ακόμα και αν δεν προσφέρει κάποια λειτουργία.
Για τη καλύτερη οργάνωση του κώδικά είναι καλύτερο όλα τα υπόλοιπα αρχεία για το πρόσθετο να είναι σε φακέλους, όσο πιο καλά οργανωμένα τόσο πιο εύκολη θα είναι η ανάγνωση του κώδικα όπως και η επέκταση του.

2.1. Hooks

Τα πρόσθετα γράφονται κατά βάση σε PHP, οι μόνες σελίδες πιθανόν να είναι σε HTML είναι αυτές που θα εμφανίζονται στο Dashboard. Αλλά υπάρχουν και κάποιες μέθοδοι PHP που μπορούν να χρησιμοποιηθούν στο περιβάλλον του WordPress που λέγονται Hooks και χωρίζονται σε filters και actions.
Τα Hooks είναι ένας τρόπος να αλληλοεπιδρά ή και να επεξεργάζεται ο κώδικας ενός πρόσθετου ή και ενός θέματος το κώδικα το WordPress.
Τα Filter σου επιτρέπουν να αλλάξεις κάποιο περιεχόμενο την ώρα που φορτώνεται και να το εμφανίσει στη σελίδα.
Για να δημιουργήσουμε ένα filter φτιάχνουμε μια μέθοδο με τη διαδικασία που επιθυμούμε και μετά τη προσθέτουμε ως filter με την εντολή add_filter() και με τον ίδιο τρόπο αφαιρούμε ένα filter με τη εντολή remove_filter().
Τα actions επιτρέπουν την αλλαγή και τη προσθήκη μεθόδων του WordPress.
Λειτουργούν με παρόμοια λογική με τα filters και προστίθεται στο WordPress με την εντολή add_action() και αφαιρούνται με την remove_action().

Υπάρχουν κάποια πολύ σημαντικά Hooks τα οποία θα τρέξουν κάποια μέθοδο αν γίνει κάτι άλλο:

- register_activation_hook()

  Θα τρέξει μια μέθοδο όταν το plugin ενεργοποιηθεί, αρκετά χρήσιμο για τη προσθήκη πινάκων στη βάση και τη προσθήκη post.

- register_deactivation_hook()

  Λειτουργεί όπως και το activation αλλά η μέθοδο θα καλεστεί όταν απενεργοποιηθεί το πρόσθετο.

- register_uninstall_hook()

  Τρέχει τη μέθοδο κατά την διαγραφή του πρόσθετου. Όταν ένα πρόσθετο δημιουργεί κάποιους πίνακες στη βάση θα πρέπει όταν ο χρήστης το διαγράψει να μην αφήσει υπολείμματα πίσω, να διαγραφτούν δηλαδή ότι αλλαγές έκανε.

Μερικά ακόμα Shortcodes

- add_shortcode()

  Προσθέτει ένα shortcode στο WordPress. Έχει δύο παραμέτρους, η πρώτη είναι το όνομα που θα δώσουμε στο shortcode και το άλλο μια μέθοδος. Όταν προστεθεί στο dashboard του WordPress το συγκεκριμένο shortcode θα εκτελεστεί η μέθοδος. Αν η μέθοδος επιστρέφει κάτι (return) θα το εμφανίσει όπου τοποθετήσουμε το όνομα του shortcode.

- get_current_user_id()

  Επιστρέφει το ID του user που είναι συνδεδεμένος στο WordPress

- wp_insert_post();

  Προσθέτει ένα post ή ένα page στο WordPress. Η σύνταξή του αποτελείτε τρείς παραμέτρους. Η πρώτη είναι ένας πίνακας με τα στοιχεία του τι θα προστεθεί, ενώ τα άλλα δύο είναι προαιρετικά και αφορούν την εμφάνιση λάθους και το αν θα ενεργοποιηθεί το Hook για μετά την εισαγωγή.

- get_option( )

  Επιστρέφει το ID της επιλογής που επιθυμούμε. Η πρώτη παράμετρος είναι το όνομα του πεδίου που θέλουμε να πάρουμε το ID και το δεύτερο πεδίο είναι προαιρετικό και αν δώσουμε true θα επιστρέψει αν δεν υπάρχει το πεδίο που ψάχνουμε, αν δεν βάλουμε κάτι είναι από μόνο του false.

- plugin_dir_path( )

  Επιστρέφει το path μέχρι το σημείο που ξεκινάνε το σημείο που ξεκινάνε τα αρχεία του πρόσθετου.

- plugin_dir_url( )

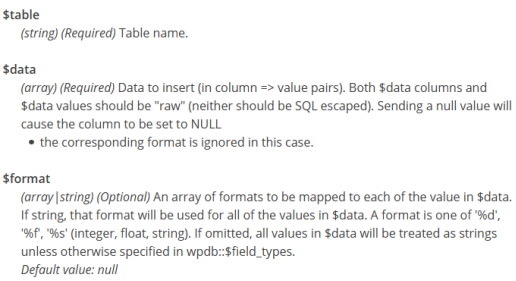
  Επιστρέφει το url μέχρι το σημείο που ξεκινάνε το σημείο που ξεκινάνε το πρόσθετο.

2.2. WPDB

Όταν γίνεται εγκατάσταση ενός WordPress θα δημιουργηθούν στη βάση κάποιοι πίνακες όπως φαίνονται στην εικόνα 3. Υπάρχουν πολλά Hooks για να εμφανίσουμε να επεξεργαστούμε ακόμα και να προσθέσουμε στη βάση του WordPress.
Μέσω της PHP το γνωστό είναι ότι για να γίνει σύνδεση με τη βάση θα πρέπει να φτιάξουμε ένα αρχείο που να το κάνει αυτό, στο οποίο θα δίνουμε στοιχεία όπως username, host, κωδικός και άλλα, και όταν θέλουμε να αλληλοεπιδράσουμε με τη βάση θα κάναμε include το αρχείο αυτό.
Τώρα όμως που δουλεύουμε μέσα στο WordPress το μόνο που χρειάζεται να κάνουμε είναι να γράφουμε την εντολή global $wpdb; και έτσι απλά μπορούμε να επικοινωνήσουμε με τη βάση του WordPress.
Αντί για το κλασσικό τρόπο που χρησιμοποιούμε στη PHP, δηλαδή να ορίσουμε το Sql Statement σε μια μεταβλητή και μετά να τη καλέσουμε με κάποια Mysql εντολή όπως *$mysql -> query($sql_statement)*, το WordPress όμως τα πράγματα είναι διαφορετικά.
Υπάρχουν έτοιμα Hooks για την εισαγωγή, για την επεξεργασία και για τη διαγραφή που μπορεί στην αρχή ναι φαίνεται περίπλοκο αλλά είναι πολύ απλά.
Στην επίσημη ιστοσελίδα του WordPress υπάρχει επεξήγηση για το κάθε Hook. Ας πάρουμε σαν παράδειγμα την εισαγωγή μια καταχώρησης σε ένα πίνακα με όνομα my_table και τα πεδία entry1 και entry2. Μας δίνεται ο παρακάτω τύπος:

wpdb->insert(string $table, array $data, array|string $format = null)

και επεξήγηση για τη κάθε παράμετρο.



$table
(string) (Required) Table name.

$data
(array) (Required) Data to insert (in column => value pairs). Both $data columns and $data values should be "raw" (neither should be SQL escaped). Sending a null value will cause the column to be set to NULL
- the corresponding format is ignored in this case.

$format
(array|string) (Optional) An array of formats to be mapped to each of the value in $data. If string, that format will be used for all of the values in $data. A format is one of '%d', '%f', '%s' (integer, float, string). If omitted, all values in $data will be treated as strings unless otherwise specified in wpdb::$field_types.
Default value: null

Εικόνα 2- Παράδειγμα Hook

Έχουμε δύο επιλογές για τις παραμέτρους μας, είτε να βάλουμε κατευθείαν μέσα στο τύπο είτε να τις ορίσουμε από πριν. Για να είναι όμως ο κωδικός πιο οργανωμένος και κατανοητός είναι καλύτερα να ορίσουμε πρώτα τις παραμέτρους και μετά να τις βάλουμε στο τύπο.
Αρχικά ο πίνακας θα οριστεί ως εξής:

$table = “my_table”;

Η παράμετρος $data είναι ένας πίνακας με της τιμές που θέλουμε να έχει η καταχώρηση και θα πρέπει να δοθεί για κάθε πεδίο ο τίτλος που έχει μέσα στη βάση και η τιμή που θέλουμε να το δώσουμε. Ο πίνακας αυτός θα έχει τέτοια μορφή:

$data = array(

'entry1' => “this is entry1”,

'entry2' => “this is entry2”

);

Τέλος η παράμετρος format δεν είναι υποχρεωτική, είναι ένας πίνακας και δηλώνει το τύπο των πεδίων που δώσαμε στο προηγούμενο πίνακα %d για integer, %f για float και %s για string. Στη συγκεκριμένη περίπτωση και τα δύο πεδία είναι κείμενο όποτε ο πίνακας θα ήταν:

$format = array(

'%s',

'%s'

);

Τώρα που έχουμε ορίσει όλες τις παραμέτρους μπορούμε να τις βάλουμε μέσα στο Hook, αφού πρώτα έχουμε κάνει τη σύνδεση με τη βάση με την εντολή

global $wpdb;

Αν έχουμε περισσότερα από ένα sql statements αρκεί να έχουμε τη παραπάνω εντολή μια φορά μέσα στο αρχείο δεν χρειάζεται να τη καλούμε κάθε φόρα.

Στη περίπτωση που έχουμε μόνο τις υποχρεωτικές παραμέτρους θα έχομε:

wpdb->insert($table,$data);

Ενώ αν θέλουμε να ορίσουμε και τους τύπους των πεδίων θα έχουμε:

wpdb->insert($table,$data,$format);

Για τη αναναίωση ενός πεδίου έχουμε το παρακάτω:

wpdb->update(string $table, array $data, array $where, array|string $format = null, array|string $where_format = null)

Τα πεδία $table, $data, $format είναι τα ίδια με την εισαγωγή. Έχουμε όμως δύο καινούρια το $where και το $where_format.

Το πεδίο $where προσδιορίζει σε ποια καταχώρηση θα γίνει η επεξεργασία και είναι και αυτό όπως και το $data ένας πίνακας διότι μπορούμε να έχουμε περισσότερους από έναν περιορισμό όπως και σε ένα SQL query θα μπορούσαμε να πούμε κάτι τέτοιο:

UPDATE my_table SET (entry1 = “this is entry 1” , entry2 = “this is entry 2”) WHERE entry3 = 5 AND entry entry4 = 2;

Μία τέτοια αλλαγή στην βάση του WordPress θα γίνονταν ως εξής.

Αρχικά θα δηλώναμε τις παραμέτρους όπως και πριν.

$table = “my_table”;

$data = array(

'entry1' => “this is entry1”,

'entry2' => “this is entry2”

);

Το πεδίο where θα προσδιοριζόταν παρόμοια με το data

```
$where = array(
' entry3' => 5,
' entry4' => 2
);
```

Το πεδίο format είναι και σε αυτή τη περίπτωση προαιρετικό όπως και το where_format που προσδιορίζει το τύπο των στοιχείων στο πίνακα where.

```
$format = array(
'%s',
'%s'
);
$where_format = array(
'%d',
'%d'
);
```

Αφού έχουμε δηλώσει όλες τις μεταβλητές που χρειαζόμαστε μπορούμε να τρέξουμε το query αφού πρώτα κάνουμε τη σύνδεση με τη βάση αν δεν έχει γίνει ήδη μέσα στο αρχείο

```
global $wpdb;
wpdb->insert($table,$data,$where,$format,$where_format);
```

Για τη διαγραφή τα πράγματα είναι πολύ πιο απλα αφού η μέθοδος παιρνει μόνο μέχρι τρείς παραμέτρους τις οποίες έχουμε ηδη δει στις δύο προηγούμενες περιπτώσεις.

```
wpdb->delete( string $table, array $where, array|string $where_format = null )
```

Οπότε αν θέλαμε να διαγρψουμε τη καταχώρηση με entry3 ίσο με το 5 θα είχαμε:

```
$table = “my_table”;
$where = array(
' entry3' => 5
);
$where_format = array(
'%d'
);
wpdb->delete( $table,$where,$where_format )
```

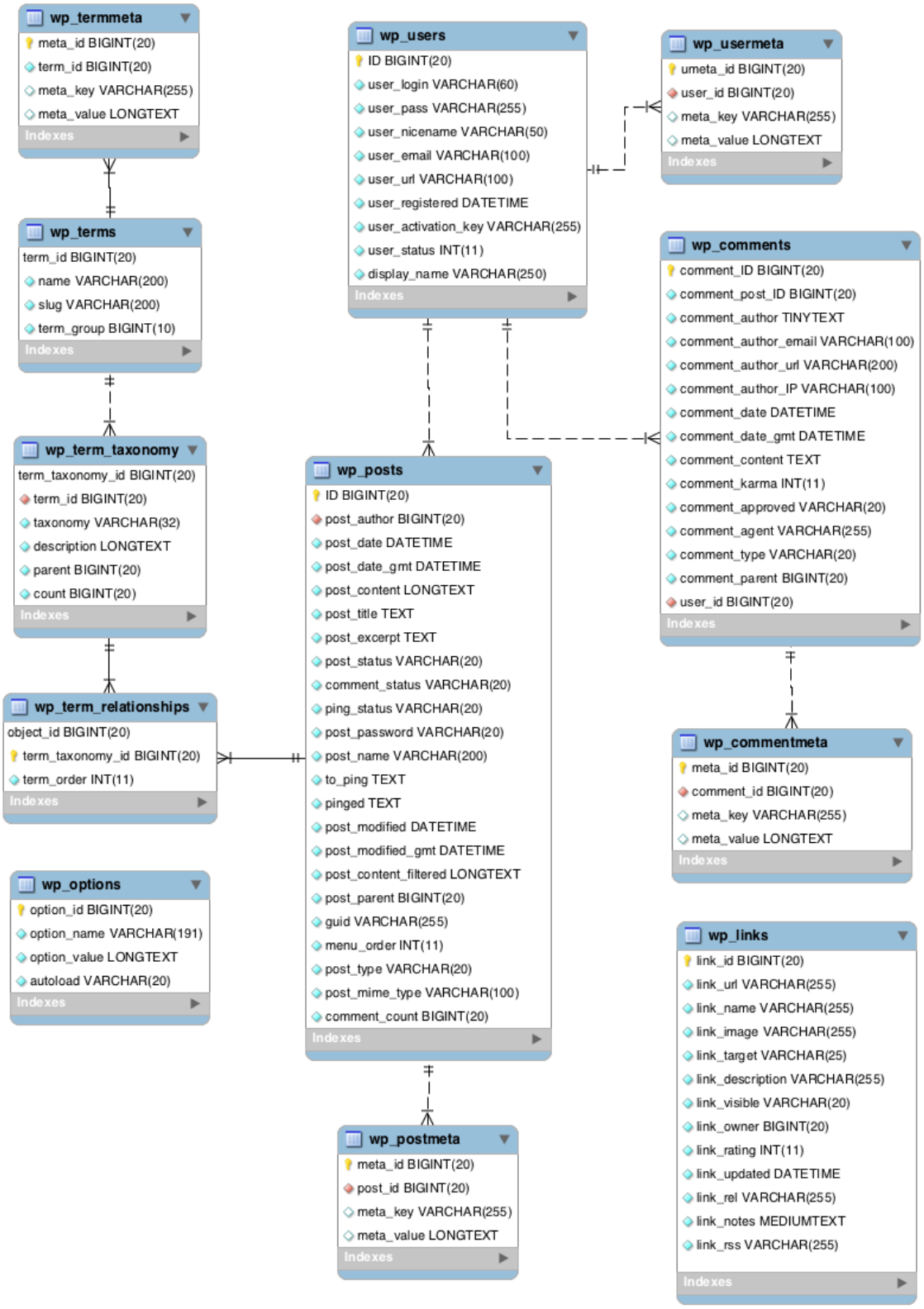
Για τη δημιουργία και τη διαγραφή νέων πινάκων χρειάζεται ο κλασσικός τρόπος αλλά με κάποιές διαφορές. Όπως και με τα Hooks θα χρειαστεί και εδώ η σύνδεση με τη βάση. Έπειτα ετοιμάζουμε το statement, δηλαδή δίνουμε σε μια PHP μεταβλητή το SQL query που θέλουμε να εκτελέσουμε.

Μετά θα χρειαστεί να path σε ένα συγκεκριμένο αρχείο μέσα στο WordPress.

```
require_once(ABSPATH . 'wp-admin/includes/upgrade.php');
```

Και τέλος θα τρέξουμε την μεταβλητή που ορίσαμε ως εξής:

```
dbDelta($sql_statemant);
```



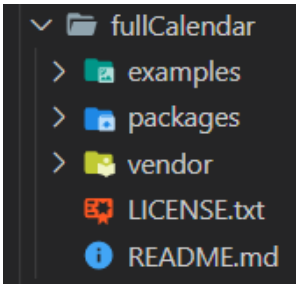
Εικόνα 3 - Πίνακες WordPress

3. Βιβλιοθήκες και Εργαλεία

Για τη δημιουργία του πρόσθετου χρησιμοποίηση τρείς πολύ χρήσιμες JavaScript βιβλιοθήκες, χωρίς να μετράω σε αυτές την jQuery. Για να προστεθεί μια βιβλιοθήκη JavaScript στο κώδικα υπάρχουν δύο τρόποι. Είτε να κατεβάσουμε τα αρχεία της βιβλιοθήκης και να τα έχουμε μέσα στο φάκελο με τα αρχεία. Είτε να καλούμε μέσω HTML από το σύνδεσμο που θα δίνεται από τη ίδια τη βιβλιοθήκη.
Η jQuery είναι ίσως και η πιο γνωστή βιβλιοθήκη JavaScript που υπάρχει. Βοηθάει στο να γίνονται περισσότερες λειτουργίες με όσο το δυνατόν λιγότερο κώδικα.

3.1. FullCalendar

Το FullCalendar είναι μια πολύ γνωστή βιβλιοθήκη για την εμφάνιση ημερολογίου με πολλές διαφοροποιήσεις. Οι περισσότερες λειτουργίες περιέχονται στην δωρεάν έκδοση αλλά υπάρχει και Premium με περισσότερες δυνατότητες.
Υπάρχει δυνατότητα για ενσωμάτωση μέσω συνδέσμου ή και εισαγωγή της βιβλιοθήκες στα αρχεία του κώδικα. Στο πρόσθετο επέλεξα να κατεβάσω τη βιβλιοθήκη να τη προσθέσω στο φάκελο Includes.



Εικόνα 4 - Αρχεία Full Calendar

Το fullCalendar για να τυπώσει events πρέπει να του δοθούν σε μορφή JSON το οποίο μπορεί να γίνει εύκολα μέσω της PHP.
Εκτός από το κλασσικό τύπο ημερολογίου που προσφέρει υπάρχει και η επιλογή εμφάνισης των εκδηλώσεων σε λίστα ανά μέρα, βδομάδα, μήνα και χρόνο. Η χρήση λίστας είναι πολύ χρήσιμη σε πολλές περιπτώσεις όπως για παράδειγμα της προβολή μιας εξεταστικής μαθημάτων.
Υπάρχει αναλυτικά documentation για τη συγκεκριμένη βιβλιοθήκη στην επίσημη ιστοσελίδα του fullCalender [11].

3.2. Datatables

Η βιβλιοθήκη Datatables σχετίζεται με την εμφάνιση ενός πίνακα, τη σειρά και τη ποσότητα που θα εμφανίζονται το στοιχεία και την αναζήτηση κάποιοί στοιχείου.

```
<!-- pagination on table -->
<link rel="stylesheet" href="https://cdn.datatables.net/1.10.21/css/jquery.dataTables.min.css">
<script src="https://cdn.datatables.net/1.10.21/js/jquery.dataTables.min.js"></script>
```

Εικόνα 5 - Αναφορά σε Datatables library

Όταν ένας πίνακας στοιχείων περιέχει μεγάλο αριθμό στοιχείων πρέπει να υπάρχει κάποια ταξινόμηση. Αυτή η βιβλιοθήκη εκτός από τη μαζεμένη εμφάνιση που δίνει στο πίνακα εμφανίζει και μόνο 10, 50
ή 100 από τα στοιχεία και υπάρχει η δυνατότητα για πλοήγηση σε επόμενες σελίδες όπως και η αλλαγή της ταξινόμησης. Ένα από τα πιο σημαντικά features που προσφέρει είναι η δυνατότητα γρήγορης αναζήτησης

μέσα στο πίνακα.
Η δημιουργία ενός μέσου αναζήτησης είναι αρκετά περίπλοκη και ειδικά αν θέλουμε να επιστρέφει άμεσα το αποτέλεσμα. Όταν έχουμε ένα πίνακα με περισσότερα από 100 καταχωρήσεις αν δεν υπάρχει δυνατότητα αναζήτησης μέσα σε αυτόν θα ήταν αδύνατο για το χρήστη να βρει αυτό που θέλει.

3.3. iCalendar

Το iCalendar είναι ένα είδος αρχείου που πολλά μέσα όπως το Google Calendar και το Apple Calendar μπορούν να το μεταφράσουν σε ένα ή πολλά events. Ένα παράδειγμα με ένα event γραμμένο σε iCalendar:

BEGIN:VEVENT
DTSTART:20150214T111500Z
DTEND:20150214T125000Z
DTSTAMP:20201217T195001Z
UID:lcvad3aq4m2jfpv1bmiihtnvvo@google.com
CREATED:20150819T000731Z
DESCRIPTION:
LAST-MODIFIED:20150819T000732Z
LOCATION:SKG
SEQUENCE:1
STATUS:TENTATIVE
SUMMARY:Leaving to LGW
TRANSP:OPAQUE
END:VEVENT

Το συγκεκριμένο κομμάτι το έχω πάρει από το αρχείο iCalendar που έκανα export από το google calendar και αντιπροσωπεύει ένα events που είχα εγώ εισάγει από το κινητό μου στο ημερολόγιο.
Αν πάρουμε την ημερομηνία *20150214T111500Z* βλέπουμε ότι το συγκεκριμένο event ήταν στις 14/02/2015 (***20150214****T111500Z*) στις 11:15 (*20150214T****111500****Z)*. Με την ίδια λογική μπορούμε να μεταφράσουμε και τις υπόλοιπες ημερομηνίες. Μπορούμε επίσης να διαβάσουμε το τίτλο (SUMMARY), την επεξήγηση (DESCRIPTION) και τη τοποθεσία (LOCATION).
Αυτό που θέλουμε από το πρόσθετο είναι να μπορεί να μεταφράσει αυτό το αρχείο και να το εισάγει στη βάση. Η ημερομηνία για παράδειγμα στη βάση πρέπει να εισαχθεί με τη μορφή DATETIME η οποία είναι YYYY-MM-DD HH:MI:SS.

3.4. VS Code

Πιστεύω είναι πολύ σημαντικό όταν γράφουμε κώδικα να είναι ευκολά αναγνώσιμος για αυτό και είναι απαραίτητη η χρήση ενός text editor που επισημαίνει με διαφορετικό χρώμα το κάθε είδους εντολή όπως και η αναγνώριση λάθους.
Τα δύο πιο γνωστά text editor για κώδικα είναι το sublime text και το vs code (visual studio code). Και τα δύο προσφέρουν παρόμοια πράγματα και πιστεύω είναι θέμα συνηθείας το ποιο χρησιμοποιεί ο καθένας.
Όλο το πρόσθετο το έχω γράψει πάνω στο vs code. Εκτός από τη επισήμανσή λαθών πολύ σημαντικό είναι και η προβολή ενός ολόκληρου φάκελου στην αριστερή μεριά του προγράμματος για τη δημιουργία μιας καλής δομής, επίσης αναγνωρίζει την κατάληξη των αρχείων ώστε να μπορεί να προβάλει με διαφορετικά χρώματα τη δομή του κώδικα ανάλογα σε τι γλώσσα είναι γραμμένα..
Το vs code έχει τη δυνατότητα εγκατάστασης διάφορων plugin για ακόμα περισσότερες λειτουργίες, ένα από τα πιο χρήσιμα είναι η επιλογή αυτόματης στοίχισης του κώδικα κάτι που κάνει το script πολύ πιο ευδιάκριτο.

3.5. Git Hub

Ένα ακόμα εργαλείο που με βοήθησε πολύ στη εργασία αυτή είναι το git hub. Αποθηκεύει το κώδικα online και κρατάει και όλες τις προηγούμενες εκδοχές σε περίπτωση που χρειαστούν. Όταν κάποιος δουλεύει πάνω σε κώδικα σε περισσότερες από μια συσκευές το Git Hub είναι πολύ πιο γρήγορο από το να πρέπει να αντιγράφεται η δουλεία σε κάποιο usb stick κάθε φορά.
Επίσης αν κάτι πάει λάθος υπάρχει πάντα η δυνατότητα να επιστρέψουμε σε μια προηγούμενη φάση του κώδικα. Για αυτό και είναι σημαντικό όταν δουλεύουμε με git hub να γίνεται commit με κάθε σημαντική αλλαγή όπως και η προσθήκη σχολίων για το τι αλλαγή κάναμε. Αυτό είναι ακόμα πιο σημαντικό όταν δουλεύουμε σε κάποια ομάδα και έχουν όλα τα μέλη access στο αρχείο στο git hub.

4. Εφαρμογή

4.1. Λειτουργία

1. ***Ανάγκη***

   Δεν είναι πάντα εύκολή η δημιουργία ενός ημερολογίου σε μια ιστοσελίδα. Ειδικά μέσα στο WordPress χωρίς γνώσεις PHP ή δυνατότητα για αγορά ενός πρόσθετου που να προσφέρει και τη εισαγωγή event και τη εμφάνισή τους σε ένα πρόγραμμα.

2. ***Χρήση***

   Η βασική χρήση του πρόσθετου "wps-my-cal" είναι η εισαγωγή, η επεξεργασία, η διαγραφή και η εμφάνιση εκδηλώσεων (που μπορεί να είναι μαθήματα, εξετάσεις κ.α.) σε πρόγραμμα. Υπάρχει η δυνατότητα εμφάνισης πολλών διαφορετικών προγραμμάτων σε διαφορετικές ή και στην ίδια σελίδα του ιστότοπου. Επίσης το προσθετό προφέρει τη δυνατότητα ταξινόμησης των event σε κατηγορίες.

4.2. Εγκατάσταση σε WordPress

1. ***Τοπικά***

   Για να εγκαταστήσουμε το plugin σε μια ιστοσελίδα WordPress τοπικά θα πρέπει αρχικά να αντιγράφει ο φάκελος "wps-my-cal" στα αρχεία του WordPress, συγκεκριμένα στο φάκελο "WordPress-name\wp-content\plugins".

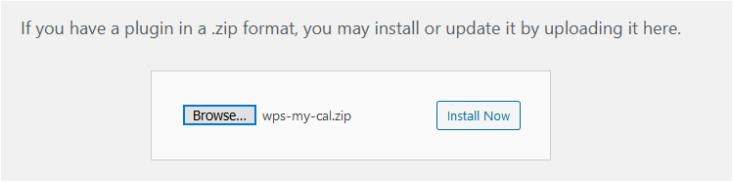
   Μετά από το WordPress Dashboard>Plugins/Πρόσθετα θα εμφανιστεί η επιλογή - όπως φαίνεται στην εικόνα.

Εικόνα 6 – MyCal στο Dashboard

   Πατώντας Activate/Ενεργοποίηση το plugin είναι έτοιμο για χρήση.

2. ***Server***

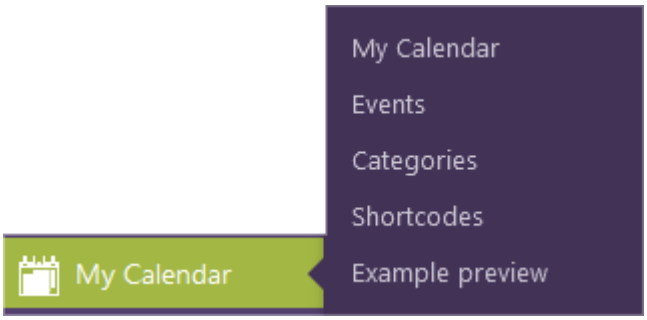
   Για να εγκαταστήσουμε το Plugin σε σέρβερ μέσω του WordPress Dashboard>Plugins επιλέγουμε Add New/Προσθήκη Νέου>Upload Plugin/Ανέβασμα , μετά πατώντας browse θα ανοίξει file manager και θα πρέπει να διαλέξουμε το φάκελο του plugin αλλά σε μορφή zip. Πατάμε Install Now/Εγκατάσταση και τέλος μετά από λίγο χρόνο θα εμφανιστεί η επιλογή Activate Plugin/Ενεργοποίηση πρόσθετου.



Εικόνα 7 - Εγκατάσταση σε server

4.3. Χρήση του Plugin

Μετά την ενεργοποίηση στο dashboard του WordPress θα εμφανιστεί μια ακόμα επιλογή με 5 tabs. Επίσης θα δημιουργηθεί μια νέα σελίδα(page) με όνομα Calendar, όπου θα έχει ένα calendar, η σελίδα αυτή υπάρχει ως παράδειγμα και μπορεί να διαγραφεί.



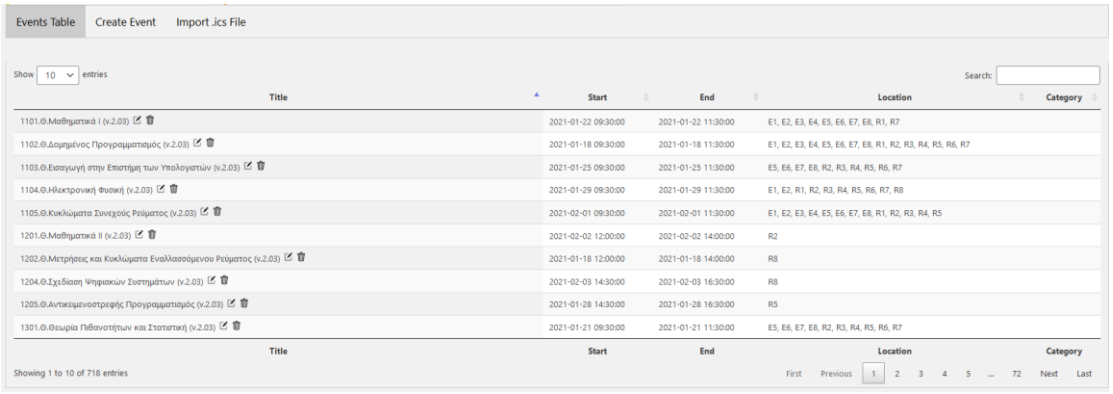
Εικόνα 8 – Το πρόσθετο στο WordPress

1. *My Calendar*

Η συγκεκριμένη σελίδα είναι εισαγωγική και έχει σκοπό να δώσει στο χρήστη κάποιες γενικές πληροφορίες για τις λειτουργίες του πρόσθετου και να τον βοηθήσει να το χρησιμοποιεί.

2. *Events*

- Events Table: Εμφάνιση όλων των event που έχουμε εισάγει. Καθώς και η δυνατότητα αναζήτησης, επεξεργασίας και διαγραφής ενός event.
- Create Event: Δημιουργία νέου event. Τα πεδία Title, Start Date, Start Time, End Date, End Time είναι υποχρεωτικά. Όλα τα υπόλοιπα είναι προαιρετικά. Επίσης κανένα πεδίο δεν είναι μοναδικό, δηλαδή δυο event μπορούν να έχουν τον ίδιο τίτλο αλλά θα αποκτήσουν διαφορετικό ID. Όλα τα πεδία μπορούν να αλλάξουν στη πρώτη καρτέλα.
- Import .ics File: Εισαγωγή πολλών event μέσω αρχείου iCalendar. Πατώντας "Browse" ανοίγει το file explorer και ο χρήστης διαλέγει ένα iCalendar αρχείο από τον υπολογιστή του και ύστερα πατάει "Upload". Μετά θα εμφανιστεί επιβεβαίωση για το πόσα event έχουν προστεθεί..



Εικόνα 9 – Εμφάνιση Events

3. *Categories*

- Categories Table: Εμφάνιση όλων των κατηγοριών που έχουμε δημιουργήσει. Καθώς και η δυνατότητα αναζήτησης επεξεργασίας και διαγραφής μιας κατηγορίας.

- Create Category: Δημιουργία νέας κατηγορίας. Ζητούνται τέσσερα πεδία: Title, Description, URL και Color. Οι επιλογές Title και Color είναι μοναδικές, αν ο χρήστης επιλέξει κάποιο που ήδη υπάρχει δεν θα γίνει η καταχώρηση και θα εμφανιστεί αντίστοιχη προειδοποίηση. Η επιλογές Description και URL είναι προαιρετικές. Αν ο χρήστης δώσει ένα URL σε μία κατηγορία, όσα event περιέχονται στη συγκριμένη κατηγορία και δεν λάβουν κάποιο URL από το χρήστη, θα χρησιμοποιήσουν το URL της κατηγορίας.

4. Shortcodes

Στα shortcodes υπάρχουν τρεις επιλογές για να αντιγραφούν, πατώντας “copy”. Οι δυο πρώτες είναι το calendar και η λίστα με όλα τα events, ενώ η τρίτη επιλογή (Create Custom) δίνει επιλογές για τη δημιουργία ενός calendar ή μιας λίστας με όλα ή συγκεκριμένα categories όπως και συγκεκριμένο ύψος και πλάτος. Και στις 3 επιλογές υπάρχει το κουμπί “copy” από τη δεξιά μεριά και το shortcode μπορεί να αντιγραφεί σε οποιαδήποτε σελίδα wordpress για να εμφανιστεί το αντίστοιχο calendar.

Εφόσον το shortcode έχει επικολληθεί σε κάποια σελίδα μπορεί να γίνει και χειροκίνητα κάποια αλλαγή (όπως πχ calendarwidth=”70%” σε calendarwidth=”75%” για να μεγαλώσει λίγο το πλάτος). Υπάρχουν τέσσερα (4) διαφορετικά shortcodes, τα οποία είναι “view-calendar”, “view-calendarlist” , “view-calendar-choose-cat” και “view-calendarlist-choose-cat”.

Shortcode attributes

- calendarwidth=”X”

Ρύθμιση του πλάτους του ημερολογίου.

Μπορεί να πάρει τιμές “1-100%” αλλά αν ο χρήστης δεν θέλει κάτι πιο ακριβές μπορεί να εισάγει και τιμές σε pixels (πχ “50px”).

*[view-calendar **calendarwidth="70%"** calendarheight="70%" locale="en-gb"]*

Το Attribute αυτό χρησιμοποιείτε στα Shortcodes

- view-calendar
- view-calendarlist
- view-calendar-choose-cat
- view-calendarlist-choose-cat

- caledarheight=”X”

Ρύθμιση του ύψους του ημερολογίου.

Μπορεί να πάρει τιμές “1-100%” αλλά αν ο χρήστης δεν θέλει κάτι πιο ακριβές μπορεί να εισάγει και τιμές σε pixels (πχ “50px”).

*[view-calendar calendarwidth="70%" **calendarheight="70%"** locale="en-gb"]*

Το Attribute αυτό χρησιμοποιείτε στα Shortcodes

- view-calendar
- view-calendarlist
- view-calendar-choose-cat
- view-calendarlist-choose-cat

- locale=”X”

Ρύθμιση της γλώσσας του ημερολογίου. Μέσω του shortcode generator δίνονται μόνο 2 επιλογές (gr και en-gb) αλλά μπορεί ο χρήστης να το αλλάξει σε ότι επιθυμεί.

[view-calendar calendarwidth="70%" calendarheight="70%" ***locale="en-gb"****]*

Το Attribute αυτό χρησιμοποιείτε στα Shortcodes

- view-calendar
- view-calendarlist
- view-calendar-choose-cat
- view-calendarlist-choose-cat

- view=”X”

Το συγκεκριμένο attribute αφορά τις λίστες. Κάθε λίστα μπορεί να έχει μέχρι και 4 διαφορετικές επιλογές την εμφάνιση: Day, Week, Month και Year. Στο ShortCode Genarator ο χρήστης έχει τη δυνατότητα να επιλέξει ποιες από αυτές τις επιλογές θα υπάρχουν.

[view-calendarlist calendarwidth="50%" calendarheight="50%" ***view = "DWY"*** *locale="el"]*

Το Attribute αυτό χρησιμοποιείτε στα Shortcodes

- view-calendarlist
- view-calendarlist-choose-cat

- countcats=”X”

Όταν χρήστης έχει δημιουργήσει κατηγορίες για τα event θα έχει τη δυνατότητα να προσθέσει calendar ή λίστες με events από συγκεκριμένες κατηγορίες. Το at-tribute countcats υποδεικνύει πόσες κατηγορίες θα υπάρχουν.

[view-calendar-choose-cat calendarwidth="50%" calendarheight="50%" ***countcats = "3"*** *cat1 = "category-1" cat2 = "category-2" cat3 = "category-3" locale="el"]*

Το Attribute αυτό χρησιμοποιείτε στα Shortcodes

- view-calendar-choose-cat
- view-calendarlist-choose-cat

- catX = “X”

Ακριβώς μετά το attribute countcats θα ακολούθουν X attributes, όπου X ο αριθ-μός των κατηγοριών, όπου το καθένα θα δίνει το slug της κατηγορίας (το slug δη-μιουργείτε αυτόματα όταν δημιουργηθεί μια κατηγορία). Οπότε θα εμφανίζετε κάτι τέτοιο countcats=”3” cat1=”category-1” cat2=”category-2” cat3=”category-3”.

[view-calendar-choose-cat calendarwidth="50%" calendarheight="50%" countcats = "3" ***cat1 = "category-1" cat2 = "category-2" cat3 = "category-3"*** *locale="el"]*

Το Attribute αυτό χρησιμοποιείτε στα Shortcodes

- view-calendar-choose-cat
- view-calendarlist-choose-cat

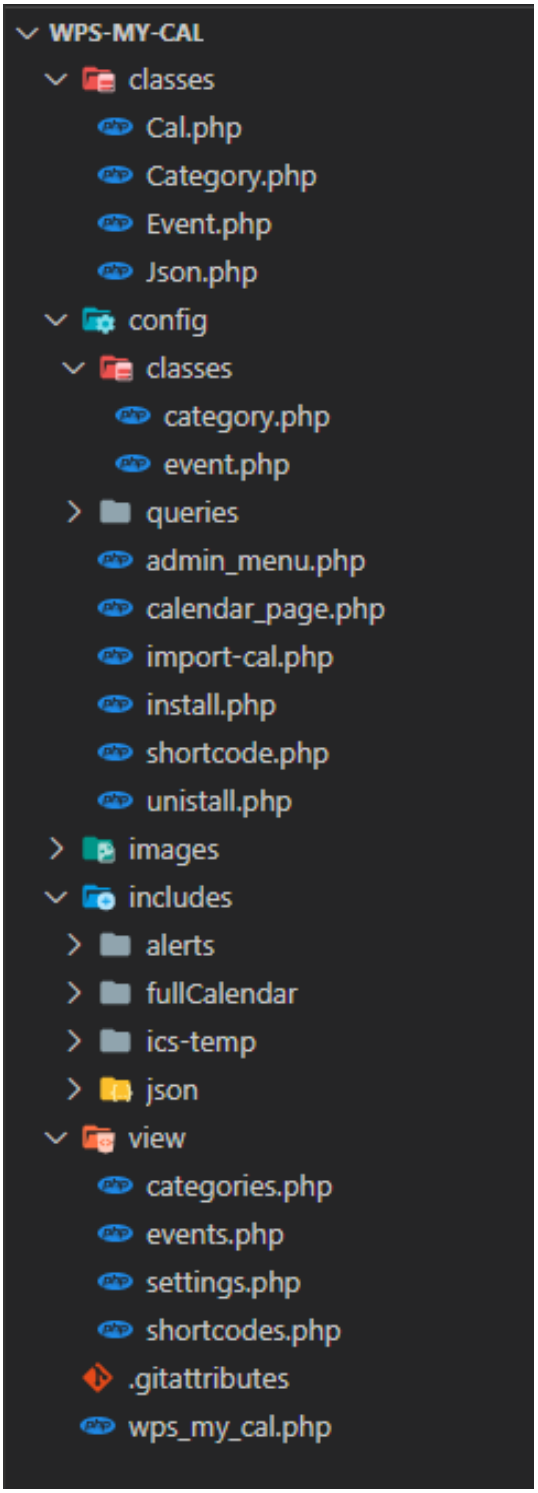
5. Example Preview

Σε αυτή τη σελίδα υπάρχουν δύο παραδείγματα, ένα με calendar και ένα με λίστα. Και τα 2 περιέχουν όλα τα events έχουν προστεθεί.

4.4. Δομή

1. Γενικά

Έκτος από τους φακέλους που θα αναλυθούν παρακάτω το main file του plugin (wps_my_cal.php), ίδιο όνομα με το φάκελο του plugin, περιέχει το comment που δίνει τη δυνατότα στο WordPress να διαβάσει το αχρείο σαν Plugin. Το συγκεκριμένο αρχείο περιέχει όλες τις εντολές include που “ενώνουν” τις τέσσερις(4) σελίδες view με τα υπόλοιπα αρχεία του plugin. Επίσης το αρχείο καλεί και τις λειτουργίες ενεργοποίησης/απενεργοποίησης καθώς και την εισαγωγή ShortCodes στο WordPress.



Εικόνα 10 - Δομή Πρόσθετου

2. *Classes*

Στο φάκελο classes υπάρχουν 4 classes του plugin. Τα 4 αυτά αρχεία αποτελούνται μόνο από PHP.

- Cal – Για τη ανάγνωση ICS file. Η κλάση Cal καλείτε από το config/import-cal. Με τη χρήση regular expressions ‘’καθαρίζει’’ το iCalendar αρχείο και επιστρέφει τις πληροφορίες που θα προστεθούν στη βάση.
- Category – Για τη δημιουργία, την επεξεργασία και τη διαγραφή κατηγορίας από τη βάση. Η κλάση Category καλείτε από το config/classes/category.php και περιέχει τρείς μεθόδους που αφορούν τις κατηγορίες. Όλες οι μέθοδοι επικοινωνούν με τη βάση με WordPress hooks.
- Event – Για τη δημιουργία, την επεξεργασία και τη διαγραφή εκδήλωσης από τη βάση. Η κλάση Event καλείτε από το config/classes/event.php και περιέχει οκτώ μεθόδους που αφορούν τα events.

  Οι μέθοδοι για την εισαγωγή και την επεξεργασία Event διαφοροποιούνται με το αν στο event θέλουμε να ορίσουμε και κατηγορία όπως και αν είναι το event μέσω του iCalendar. Επίσης υπάρχει μια επιπλέον μέθοδο για επεξεργασία για όταν ο χρήστης διαγράψει μια κατηγορία που αφαίρει τη συγκεκριμένη κατηγορία από όσα event την έχουν σαν attribute.
- JSON – Για τη δημιουργία, την επεξεργασία και τη διαγραφή JSON files. Σε αυτή τη κλάση υπάρχουν τέσσερις μέθοδοι. Οι δύο updateAllJSON() και updateAllCategoriesJSON() δημιουργούν αρχεία JSON και τα γεμίσουν τα events από τη βάση ώστε να μπορεί να τα τυπώσει το full calendar και καλούνται αρκετές φορές μέσα στο plugin μετά από κάθε νέα εισαγωγή, επεξεργασία και διαγραφή. Ενώ οι μέθοδοι createOneEventJSON($event_id) και createOneCatJSON($cat_id) δημιουργούν ένα JSON αρχείο το καθένα και γράφουν μέσα ένα event και ένα category από τη βάση ανάλογα με το ID που θα τους δοθεί, καλούνται από τα config/queries/fetch_event.php και config/queries/fetch_category.php αντιστοιχία κάθε φορά που ο χρήστης επιλέγει να επεξεργαστεί ένα event ή μία κατηγορία.

3. *Config*

Στο φάκελο config έξι αρχεία PHP και δύο φάκελοι, Classes και Queries, με δύο αρχεία PHP ο καθένας.

- install/uninstall

  Τα δύο αυτά αρχεία περιέχουν μία μέθοδο το καθένα όπου καλούνται από το wps_my_cal.php όταν ενεργοποιείται το plugin και όταν διαγράφεται. Και αυτό που κάνουν είναι να δημιουργούν και να διαγράφουν δύο πίνακες μέσα στη βάση του WordPress.
- calendar_page

  Αποτελείτε από δύο μεθόδους μία για την δημιουργία της σελίδας “My Calendar” και μία για τη διαγραφή της. Οι μέθοδοι καλούνται από το wps_my_cal.php με την ενεργοποίηση και τη διαγραφή του plugin. Η σελίδα “My Calendar” περιέχει ένα shortcode με όλα τα event ([view-calendar calendarwidth="70%" calendarheight="70%" locale="en-gb"]) και υπάρχει για βοηθήσει το χρήστη να κατανοήσει πως λειτουργεί το plugin.
- admin_menu

  Περιέχει τη μέθοδο my_cal_options_panel() η οποία δημιούργει μέσα στο WordPress Dashboard το tab για το plugin και τα τέσσερα sub-pages του (Events/Categories/Shortcodes/Example Preview). Όπως και τη μέθοδο wps_cal_func() η οποία προσθέτει κείμενο στην αρχική σελίδα του plugin.
- import_cal

  Στο αρχείο import_cal.php βρίσκονται οι ενέργειες που θα γίνουν όταν χρήστης πατήσεις το κουμπί “upload” στην υποσελίδα Events>Import .ics file. Αρχικά θα γίνει έλεγχος αν το αρχείο που επιλέχθηκε

είναι πράγματι iCalendar, αν όχι θα επιστραφεί το αντίστοιχο σφάλμα. Όταν το αρχείο είναι σωστό θα δημιουργηθεί ένα προσωρινό αρχείο $file στο οποίο θα αποθηκευτεί μέσα το περιεχόμενο του .ics. Έπειτα θα οριστεί μια μεταβλητή $iCal που θα αντιπροσωπεύει ένα νέο αντικείμενο iCal (classes>Cal.php) με attribute το αρχείο που δημιουργήθηκε το οποίο μετά από αυτό διαγράφεται. Μετά η μέθοδος eventsByDate() επιστρέφει ένα-ένα τα events και η μέθοδος addEventFromICS() τα εισάγει στη βάση. Τέλος θα επιστραφεί μήνυμα με το πόσα events προστέθηκαν.

- shortcode

  Στο αρχείο αυτό υπάρχουν τέσσερις μέθοδοι: show_calendar($atts), show_calendar_list($atts), show_calendar_choose($atts) και show_calendar_list_choose($atts). Και οι τέσσερις επιστρέφουν το ίδιο

  return " "; . Η κάθε μία από αυτές τις μεθόδους καλείτε στο wps_my_cal.php μέσω του WordPress Hook add_shortcode() για τη δημιουργία των Shortcodes. Όλα επιστρέφουν ένα fullcalendar αλλά με διαφοροποιήσεις κάθε φορά.

- Classes
  - event

    Στο event.php περιέχονται τρείς ενεργείς σε σχέση με τα Events. Όταν ο χρήστης προσθέσει ένα καινούριο event όπως και όταν επιλέξει να το επεξεργαστεί και να το διαγράψει. Σε όλες τις περιπτώσεις επιστρέφεται επιβεβαιωτικό μήνυμα ή προειδοποίηση και σε κάθε αλλαγή που θα γίνει στη βάση να γίνει ανανέωση και στα αρχεία JSON όπως φαίνεται στην εικόνα.

```
// UPDATE JSON
$json_obj = new Json();
$json_obj->updateAllCategoriesJSON();
$json_obj->updateAllJSON();
// /UPDATE JSON
```

Εικόνα 11 - Ανανέωση JSON

  - category

    Οι ενέργειες που γίνονται για τις κατηγορίες μοιάζουν πολύ με αυτές που γίνονται για τα events με τη διαφορά ότι τώρα υπάρχουν περιορισμοί. Στα event επιτρέπεται δύο ή παραπάνω καταχωρήσεις να έχουν ίδιο όνομα αλλά στις κατηγορίες θέλουμε να είναι μοναδικά. Όταν ο χρήστης επιχειρήσει να προσθέσει μια νέα κατηγορία γίνεται έλεγχος στη βάση αν υπάρχει ήδη κατηγορία με αυτό το τίτλο. Αν βρεθεί τέτοια κατηγορία η καταχώρηση δεν γίνεται και επιστρέφεται αντίστοιχο μήνυμα. Αν όμως δεν βρεθεί γίνεται ο ίδιος έλεγχος και για το χρώμα της κατηγορίες διότι και αυτό θέλουμε να είναι μοναδικό. Τέλος αν και τα δύο δεν βρεθούν στη βάση θα εισαχθεί η κατηγορία και τα επιστραφεί μήνυμα επιτυχίας.

    Ο ίδιος έλεγχος θα γίνει και στην επεξεργασία μιας κατηγορίας. Ενώ στη διαγραφεί δεν χρειάζεται κάποιος έλεγχος. Παρόλο που δεν θα ελέγξουμε το πίνακα κατηγοριών κατόπιν της διαγράφης θα χρειαστεί πριν γίνει η διαγραφή να ελεγχθεί ο πίνακας Events. Αν υπάρχει ένα event με τη κατηγορία που προσπαθούμε να διαγράψουμε θα πρέπει πρώτα να αφαιρεθεί η κατηγορία από τα attributes.

    Όπως και στα event έτσι και εδώ μετά από κάθε αλλαγή στη βάση ανανεώνονται και τα JSON αρχεία όπως στην εικόνα.

- Queries

Τα αρχεία fetch_event.php και fetch_category.php περιέχουν μια POST ενέργει το καθένα. Το μόνο που κάνουν είναι όταν το χρήστης πατήσει να επεξεργαστεί ένα event ή μια κατηγορία να “στέλνει” την ID του συγκεκριμένου event ή της κατηγορίας που επέλεξε, να δημιουργεί αντικείμενο Json() και να καλεί τις μεθόδους *createOneEventJSON($id)* και *createOneCatJSON($id).* Οπότε θα

δημιουργηθεί ένα αρχείο JSON με τα στοιχεία του event ή της κατηγορίας του επιλέχθηκε.

4. Images

Στο φάκελο Images υπάρχουν τρείς εικόνες που εμφανίζονται στο plugin. Μία για το εικονίδιο που θα υπάρχει στο Dashboard του WordPress και δύο για τα κουμπιά edit και delete που εμφανίζονται στους πίνακες με τα events και τα categories δίπλα από το τίτλο κάθε καταχώρησης.

5. Includes

Ο φάκελος includes αποτελείτε από τέσσερείς ΥΠΟ φακέλους. Στο φάκελο alert υπάρχουν τρία αρχεία PHP για την εμφάνιση των μηνυμάτων προειδοποίησης, επιτυχίας και λάθους στο χρήστη. Το fullCalendar είναι η JavaScript βιβλιοθήκη για την εμφάνιση του Ημερολογίου. Ο φάκελος ics-temp είναι ένας άδειος φάκελος όπου εκεί αποθηκεύεται προσωρινά το αρχείο iCalendar όταν ο χρήστης κάνει εισαγωγή μέσω αρχείου .ics. Και τέλος στο φάκελο Json αποθηκεύονται όλα τα αρχεία JSON που δημιουργούνται στο classes/Json.php.

6. View

Εδώ βρίσκονται οι τέσσερις σελίδες που βλέπει ο χρήστης. Τα αρχεία αυτά αποτελούνται από HTML, CSS, JavaScript και PHP.

- events

  Τα αρχείο χωρίζεται σε τέσσερα βασικά html div, add-ics, add-event, show-events και modal. Τα πρώτα τρία είναι χωρισμένα σε καρτέλες(tabs) και όταν ο χρήστης επιλέξει την επιλογή events από το dashboard το πρώτο που θα δει είναι το div show-events που περιέχει ένα πίνακα με όλα τα events που υπάρχουν στη βάση.

  Με τη βοήθεια της JavaScript βιβλιοθήκης datatables υπάρχει δυνατότητα αναζήτησης κάπου events όπως και εύκολη πλοήγηση σε επόμενες σελίδες όταν ο πίνακας είναι μεγάλος. Μέσα στο πίνακα, δίπλα στο τίτλο του event βρίσκονται δύο κουμπιά όπου το καθένα από αυτά βρίσκεται μέσα σε μια μικρή HTML φόρμα με κρυφό input το ID της κατηγορίας έτσι ώστε όταν ο χρήστης πατήσει είτε edit είτε delete να σταλθεί με POST το ID στο config/queries/fetch-event.php ή στο config/classes/event.php αντίστοιχα.

  Στη περίπτωση του edit αφού σταλθεί με POST το id και δημιουργηθεί το JSON αρχείο oneEvent.json θα δοθεί στο div modal, το οποίο είναι hidden, ένα CSS class έτσι ώστε να μην είναι κρυφό (*$(".popup-overlay, .popup-content").addClass("active");*). Μαζί με την αλλαγή στη CSS του modal γίνεται και ανάγνωση του oneEvent.json και εμφανίζονται στο modal τα στοιχεία του event που επιλέχθηκε και έχει ο χρήστης τότε τη δυνατότα να κάνει όποιες αλλαγές θέλει.

  Στη δεύτερη καρτέλα βρίσκεται το div add-event το οποίο περιέχει μια απλή φόρμα για την εισαγωγή ενός νέου event. Με τη χρήση της JavaScript βιβλιοθήκης datepicker γίνται πιο κατανοητά η εισαγωγή ημερομηνίας και για την επιλογή κατηγορίας πραγματοποιείται με PHP ένα sql statement για να επιστραφούν όλες οι καταχωρημένες κατηγορίες σε μορφή dropdown menu.

  Τέλος στο τρίτο tab είναι το div add-ics όπου με άλλο μια POST φόρμα μέσω του config/import-cal.php μπορεί ο χρήστης να ανεβάσει το iCalendar αρχείο που επιθυμεί.

- categories

  Το αρχείο categories έχει την ίδια δομή με το events αλλά με λιγότερα text inputs στις φόρμες και χωρίς Τρίτη καρτέλα. Αντί για το oneEvent.json υπάρχει το oneCategory.json.

- shortcodes

  Όλα τα shortcodes δημιουργούνται από τις μεθόδους στο config/shortcodes.php ενώ η σελίδα αυτή είναι καθαρά για το χρήστη για να μη χρειάζεται να ξέρει πως λειτουργούν τα shortcodes που έχω φτιάξει αλλά να μπορεί να τα εισάγει ευκολά σε όποιο σημείο θέλει στη σελίδα του. Υπάρχουν σύνολο τρείς επιλογές που μπορεί να κάνει αυτόματα αντιγραφή πατώντας copy, το όποιό γίνεται με τη μέθοδο copyToClipboard() σε JavaScript.

  Οι δύο πρώτες επιλογές για αντιγραφή είναι στατικές και αντιστοιχούν σε calendar με όλα τα events και list με όλα τα events.

  Η Τρίτη επιλογή είναι η πιο σημαντική. Υπάρχουν πριν την επιλογή για αντιγραφή ορισμένες ρυθμίσεις οι οποίες αλλάζουν δυναμικά το shortcode όσο τις αλλάζει ο χρήστης.

  Άρα μπορεί ο καθένας να φτιάξει ένα shortcode όπως το θέλει, calendar ή λίστα, συγκεκριμένο μέγεθος, με όλα τα events ή συγκεκριμένα categories, ελληνικά ή αγγλικά και στη περίπτωση της λίστας μπορεί να διαλέξει συγκεκριμένα view ανάμενα σε day, week , month και year.

```
// Copy
function copyToClipboard(element) {
    var $temp = $("<input>");
    $("body").append($temp);
    $temp.val($(element).text()).select();
    document.execCommand("copy");
    $temp.remove();
}
```

Εικόνα 12 - Μέθοδος για αντιγραφή

- preview

  Η τελευταία σελίδα δεν είναι τίποτα παραπάνω από δύο παραδείγματα ενός calendar και μιας λίστας για να πάρει ο χρήστης μια γεύσή για το πως φαίνονται.

4.5. Στη Πράξη

1. Εγκατάσταση/Διαγραφή

Τη πρώτη φορά που θα ενεργοποιηθεί (activate) το plugin θα εκτελεστούν τρείς μέθοδοι, δυο μέσω του WordPress Hook *register_activation_hook()* και μία μέσω του *add_action()*. Αρχικά θα δημιουργηθούν δύο πίνακες στη ήδη υπάρχουσα βάση του WordPress.

```
// create table categories
$create_categories = "CREATE TABLE $categories_table (
            id int(11) NOT NULL auto_increment,
            title varchar(256) NOT NULL,
            slug varchar(256) NOT NULL,
            description text,
            color text NOT NULL,
            url varchar(255),
            PRIMARY KEY (id)
        )$charset_collate;";

// create tables events with foreign key to categories
$create_events = "CREATE TABLE $events_table (
            id int(11) NOT NULL auto_increment,
            uid varchar(255),
            title varchar(256) NOT NULL,
            start datetime NOT NULL,
            end datetime NOT NULL,
            description text,
            location text,
            status varchar(255),
            isReccurent BOOLEAN,
            created datetime,
            updated datetime,
            category_id int(11),
            url varchar(255),
            PRIMARY KEY (id),
            FOREIGN KEY (category_id) REFERENCES  $categories_table(id)
        )$charset_collate;";
```

Εικόνα 13 - Δημιουργία πινάκων

Όπως φαίνεται και στην εικόνα 13 πρώτα θα δημιουργηθεί ο πίνακας categories και μετά ο πίνακας events επειδή χρησιμοποιεί ξένο κλειδί (foreign key) id από τον categories. Ο τίτλος των δύο πινάκων εξαρτάται από τη σελίδα που θα εγκατασταθεί το plugin, θα χρησιμοποιήσει ως πρώτο συνθετικό το prefix της σελίδας. Όταν γίνεται εγκατάσταση μιας σελίδας WordPress ζητείται από τον χρήστη να ορίσει ένα prefix (wp_, w_ ή ότι άλλο θέλει). Αν δηλαδή το prefix είναι wp_ οι πίνακες θα δημιουργηθούν ως wp_wps_categories και wp_wps_events.

Το άλλο hook ενεργοποίησης θα δημιουργήσει μία σελίδα (page) στο WordPress. Η σελίδα αυτή θα έχει τίτλο 'My Calendar' και θα περιέχει μέσα ένα shortcode το οποίο θα εμφανίζει ένα calendar με όλα τα events που υπάρχουν στη βάση. Άρα τη πρώτη φορά που θα δημιουργηθεί θα είναι άδειο.

```php
// Create post object
$my_post = array(
    'post_title'    => 'My Calendar',
    'post_content'  => '[view-calendar calendarwidth="70%" calendarheight="70%" locale="en-gb"]',
    'post_status'   => 'publish',
    'post_author'   => get_current_user_id(),
    'post_type'     => 'page',
);
```

Εικόνα 14 - Δημιουργία Page

Αν το plugin απενεργοποιηθεί και δεν διαγραφεί οι πίνακες θα παραμείνουν στη βάση αλλά η σελίδα θα διαγραφεί λόγω του *register_deactivation_hook()*.. Για να διαγραφούν οι πίνακες θα πρέπει να διαγραφεί τελείως το plugin άρα να καλεστεί το *register_uninstall_hook().*

Εκτός από τα activation hooks όταν ενεργοποιηθεί το plugin θα εκτελεστεί και μια ακόμα μέθοδος λόγω του add_action και θα δημιουργήσει το menu στο WordPress dashboard και τέσσερεις submenu σελίδες.

```php
add_menu_page(
    'My Calender page title', //string $page_title
    'My Calendar', //string $menu_title
    'manage_options', //string $capability
    'cal-options', //string $menu_slug
    'wps_cal_func', //callable $function = ''
    plugin_dir_url(__FILE__) . '../images/cal-icon-3.png' // string $icon_url = ''
);

add_submenu_page(
    'cal-options', //string $parent_slug
    'url page title', //string $page_title
    'Events', //string $menu_title
    'manage_options', //string $capability
    plugin_dir_path(__DIR__) . 'view/events.php', //string $menu_slug
    null //callable $function = ''
);
```

Εικόνα 15 - Δημιουργία Menu

2. *Προσθήκη Event και Category*

Η διαδικασία για τη προσθήκη ενός event και αυτή για τη προσθήκη κατηγορίας έχουν αρκετά κοινά αλλά σε κάποια σημεία διαφέρουν αρκετά. Και στις δυο περιπτώσεις το UI είναι αρκετά απλό και έχει δημιουργηθεί με HTML φόρμα.

Στη εισαγωγή Event υπάρχουν περισσότερα πεδία και ένα από αυτά "τραβάει" και δεδομένα από τη βάση για την επιλογή κατηγορίας. Αν δεν έχει εισαχθεί κάποια κατηγορία ακόμα η μόνη επιλογή που θα έχει ο χρήσης θα είναι μια παύλα.

Ο τίτλος είναι υποχρεωτικός και στις δύο περιπτώσεις ενώ το event έχει και σαν υποχρεωτικά πεδία τις ημερομηνίες και τις ώρες. Όλα αυτά τα παιδία γίνονται υποχρεωτικά με τη χρήση του attribute "required" στα input της φόρμας. Εκτός όμως από τα υποχρεωτικά υπάρχουν και άλλα περιορισμοί που αποτρέπουν τα event και τα categories από το να εισαχθούν. Στα events ο μόνος περιορισμός έχει να κάνει με τις

ημερομηνίες και τις ώρες, δηλαδή δεν επιτρέπεται στο χρήστη να έχει την αρχή ενός event μετά το τέλος του. Αυτός ο έλεγχος γίνεται αφού σταλθούν τα στοιχεία με POST και ο έλεγχος γίνεται με μια απλή αφαίρεση.

```php
// combine start/end date and time to "Y-m-d HH:MM:ss" format
$start = $startDate . " " . $startTime;
$end = $endDate . " " . $endTime;
//array_push($warnings, $start);
//array_push($warnings, $end);

// check if start datetime is before end datetime
$startCount = preg_replace('/[^0-9]/', '', $start);
$endCount = preg_replace('/[^0-9]/', '', $end);
$duration = $endCount - $startCount;
if ($duration < 0) {
    array_push($errors, "Start Date/Time must be before End Date/Time");
}
```

Εικόνα 16 - Έλεγχος ημερομηνίας και ώρας

Αρχικά θα γίνει η ημερομηνία έναρξης με την ώρα ένα νούμερο χρησιμοποιώντας regular expression για να αφαιρεθεί οτιδήποτε δεν είναι αριθμός, δηλαδή τα σύμβολα “/” και “:”, έπειτα θα γίνει το ίδιο και για την ημερομηνία και ώρα λήξης. Αν η διαφορά της έναρξης από την λήξη είναι μικρότερη με το μηδέν η καταχώρηση δεν θα γίνει και θα εμφανιστεί μήνυμα σφάλματος.

```php
public function addCategory($catTitle, $catSlug, $catDesc, $catUrl, $catColor)
{

    global $wpdb;
    $categories_table = $wpdb->prefix . 'wps_mc_categories';
    $catData = array(
        'title' => $catTitle,
        'slug' => $catSlug,
        'description' => $catDesc,
        'url' => $catUrl,
        'color' => $catColor
    );

    $wpdb->insert($categories_table, $catData);

    // create json file for this category
    $target_Path = plugin_dir_path(__DIR__) . "includes/json/categorized/";
    $file = fopen($target_Path . "/" . $catTitle . ".json", 'w') or die("can't open file");
    fclose($file);
}
```

Εικόνα 18 - Μέθοδος για εισαγωγή κατηγορίας

Αν όλα είναι εντάξει θα καλεστεί είτε η μέθοδος addEventWithoutCat() είτε η addEvent(). Υπάρχουν δύο διαφορετικές μέθοδοι για την εισαγωγή διότι αν δώσουμε τιμή null στη κατηγορία η εισαγωγή δεν γίνεται.

```php
global $wpdb;
$cat_table = $wpdb->prefix . 'wps_mc_categories';
$checkSameTitle = $wpdb->get_var("SELECT COUNT(*) FROM $cat_table WHERE title = '$catTitle'");
if ($checkSameTitle > 0) {
    array_push($errors, "Title already exist");
}
$checkSameColor = $wpdb->get_var("SELECT COUNT(*) FROM $cat_table WHERE color = '$catColor'");
if ($checkSameColor > 0) {
    array_push($errors, "Color already assigned in another category");
}
```

Εικόνα 17 - Έλεγχος ίδιου τίτλου και χρώματος στην εισαγωγή κατηγορίας

Και οι δύο μέθοδοι όμως ακολουθούν την ίδια μεθοδολογία. Χρησιμοποιούν το WordPress Hook

$wpdb->insert()

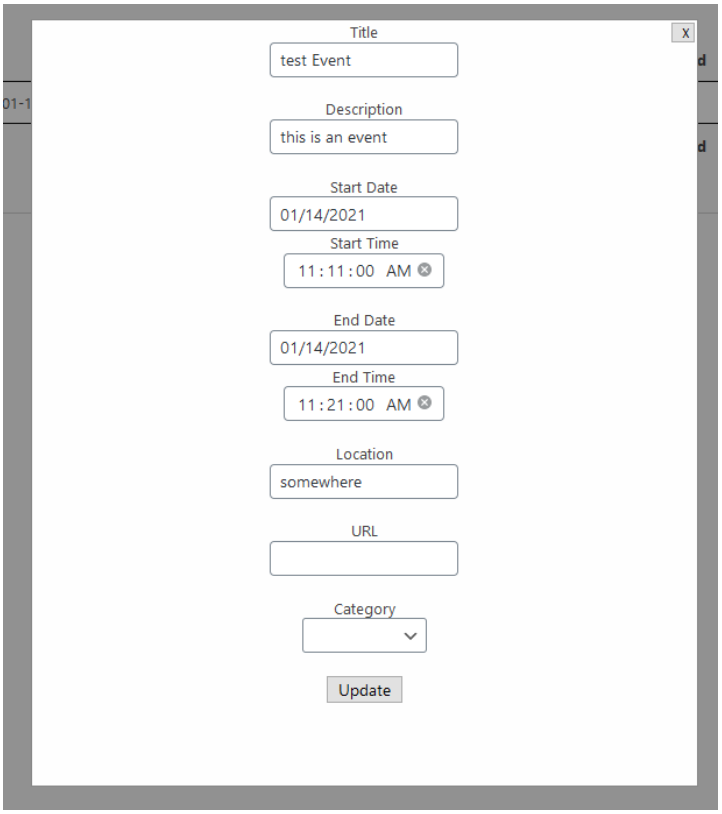
για την εισαγωγή στη βάση και η μόνη διαφορά τους είναι ότι η μία έχει κατηγορία ενώ η άλλη όχι.

Η εισαγωγή κατηγορίας έχει τους δικούς της περιορισμούς. Ένα event μπορεί να έχει ακριβώς ίδιο τίτλο με ένα άλλο ενώ μια κατηγορία θέλουμε να είναι μοναδική όπως και το χρώμα που της δίνουμε. Για αυτό και όταν τα στοιχεία που θα δοθούν θα σταλθούν στο config/category.php θα γίνει έλεγχος στη βάση και αν κάποιο από αυτά υπάρχει ήδη δεν θα γίνει η καταχώρηση και θα επιστραφεί αντίστοιχο μήνυμα σφάλματος.

Αν όλα πάνε καλά και εισαχθεί στη βάση η κατηγορία θα υπάρξει ακόμα μια ενεργεία που δεν υπήρξε στα events. Θα δημιουργηθεί ένα JSON αρχείο για τη κατηγορία.

3. Επεξεργασία και Διαγραφή

Αφού ο χρήστης έχει εισάγει events και categories έχει και τη δυνατότητα να επεξεργαστεί και να διαγράψει όποιο επιθυμεί. Δίπλα από το τίτλο των event και των κατηφοριών υπάρχουν δύο κουμπιά, το πρώτο είναι για την επεξεργασία και το δεύτερο για την διαγραφή.



Εικόνα 19 - Pop-up για επεξεργασία event

Όταν πατήσει να επεξεργαστεί κάτι θα εμφανιστεί ένα παράθυρο το οποίο είναι με HTML και είναι πολύ κοντά στη φόρμα που υπάρχει για την εισαγωγή αλλά κάποια πεδία είναι συμπληρωμένα. Όταν πατηθεί το κουμπί update η διαδικασία που θα γίνει είναι σχεδόν ίδια με της εισαγωγής αλλά η μέθοδος περιέχει μέσα το WordPress Hook *$wpdb->insert()* αλλά το *$wpdb->update().*

Η διαγραφεί και στις δύο περιπτώσεις γίνεται με το WordPress Hook *$wpdb->delete()* και είναι πολύ η πιο απλή σε σχέση με τις άλλες διαδικασίες, με μία μικρή παραλλαγή στη διαγραφή κατηγορίας. Αν προσπαθήσουμε να διαγράψουμε μια κατηγορία ενώ την έχουμε χρησιμοποιήσει σαν attribute σε κάποιο event λόγω του foreign key η διαγραφή δεν θα γίνει. Θα πρέπει πρώτα να επεξεργαστούμε τα events και μετά να διαγράψουμε τη κατηγορία. Η σειρά έχει μεγάλη σημασία.

Εικόνα 20 - Προβολή ενός event στο πίνακα

Για αυτό υπάρχει η μέθοδος *updateDeletedEvets()* που δέχεται σαν παράμετρο το ID της κατηγοριάς. Άρα όταν καλεσθεί θα αντικατστίσει τη κατηγορία με null σε όλα τα events που την έχουν. Σαν SQL query θα ήταν το "UPDATE wp_wps_events SET category_id = null WHERE category_id = deleted_category;"

```php
public function updateDeletedCatEvents($cat_id)
{
    global $wpdb;
    $events_table = $wpdb->prefix . 'wps_mc_events';

    $eventData = array(
        'category_id' => null
    );
    $where = array(
        'category_id' => $cat_id
    );
    $wpdb->update($events_table, $eventData, $where);

}
```

Εικόνα 21 - Μέθοδος για ανανέωση event πριν το σβήσιμο μιας κατηγορίας

4. JSON

Έχουν αναφερθεί αρκετές φορές αρχεία JSON και για το πώς ανανεώνονται κάθε φόρα όταν αλλάζει κάτι στη βάση. Το full calendar για να διαβάσει events και να τα τυπώσει πρέπει να είναι σε JSON μορφή.

```php
public function createOneCatJSON($cat_id)
{
    global $wpdb;
    $event_table = $wpdb->prefix . 'wps_mc_categories';

    $target_Path = plugin_dir_path(__DIR__) . "includes/json/oneJSON";

    $fp = fopen($target_Path . "/oneCat.json", 'w');
    //fwrite($fp, "[");
    $get_the_cat = "SELECT * FROM $event_table WHERE `id` = '" . $cat_id . "'";
    $row = $wpdb->get_row($get_the_cat);
    $title = $row->title;
    $slug = $row->slug;
    $description = $row->description;
    $color = $row->color;
    $url = $row->url;

    $category = '
            {
            "title": "' . $title . '",
            "slug": "' . $slug . '",
            "description": "' . $description . '",
            "color": "' . $color . '",
            "url": "' . $url . '"
            }
    ';
    fwrite($fp, $category);

    fclose($fp);
}
```

Εικόνα 22 - Μέθοδος δημιουργίας JSON για την εμφάνιση κατηγορίας στο Pop-up επεξεργασίας

Στη κλάση JSON “μεταφράζονται” τα στοιχεία από τη βάση σε JSON μορφή με τη PHP εντολή fwrite() επαναληπτικά με την εντολή foreach μέχρι να γραφτούν όλα τα event.

Τα JSON αρχεία είναι χωρισμένα, ένα αρχείο που περιέχει όλα τα events ανεξαρτήτως αν έχουν κατηγορία ή όχι και ένας φάκελος όπου είναι χωρισμένα σε κατηγορίες. Αρχικά ένα αρχείο που περιέχει όσα event δεν έχουν κατηγορία και μετά ένα αρχείο για κάθε κατηγορία το οποίο δημιουργείται όταν προστεθεί η κατηγορία. Όλα αυτά τα αρχεία ανανεώνονται μετά από κάθε νέα εισαγωγή, επεξεργασία και διαγραφή στη βάση. Ο λόγος που υπάρχουν ξεχωριστά αρχεία για τα events με κατηγορίες είναι για να μπορεί να υπάρξει calendar μόνο με κάποιες κατηγορίες, ανάλογα το shortcode που θα φτιάξει ο χρήστης.

Υπάρχουν όμως άλλα δύο αρχεία μέσα στο φάκελο JSON στο φάκελο oneJSON. Αυτά τα δύο αρχεία αλλάζουν το περιεχόμενο τους κάθε φορά που ο χρήστης θα πατήσει να επεξεργαστεί ένα event ή μια κατηγορία και θα διαβαστούν από το modal για να εμφανιστούν όλα τα στοιχεία που μπορεί να αλλάξει.

5. ShortCodes

Η δημιουργία shortcode είναι ίσως και ο πιο εύκολος τρόπος να εμφανίσεις αυτό που θέλεις εκεί που το θέλεις. Χρειάζεται μόνο δύο βήματα για να δημιουργηθεί, μία μέθοδο που να επιστρέφει το επιθυμητό από αποτέλεσμα και το WordPress Hook add_shortcode που συνδέει τη μέθοδο με μία ονομασία.

```
add_shortcode('view-calendar', 'show_calendar');

add_shortcode('view-calendarlist', 'show_calendar_list');

add_shortcode('view-calendar-choose-cat', 'show_calendar_choose');

add_shortcode('view-calendarlist-choose-cat', 'show_calendar_list_choose');
```

Εικόνα 23 - Ορισμός των Shortcodes

Και τα τέσσερα ShortCode επιστρέφουν το ίδιο. Ένα html div με το full calendar με την εντολή΄:

return “div class = ‘divCal’ id = ‘calendar’> ;

Αυτό που διαφέρει όμως κάθε φορά είναι οι επιλογές που έχουν γίνει στη JavaScript του full calendar και στα attributes. Είναι πολύ απλός ο ορισμός attributes, όπως φαίνεται στην εικόνα στο συγκεκριμένο shortcode έχουν δωθεί τρία attributes, δηλαδή μέσα στο συγκεκριμένο θα πρέπει να υπάρχει το κείμενο calendarwidth = “X” και ότι είναι το X θα είναι η τιμή της μεταβλητής $callW μέσα στη μέθοδο, και μετά τη συγκεκριμένη μεταβλητή τη χρησιμοποιώ για να ορίσω το πλάτος του div στη CSS. Μπορούμε να ορίσουμε όσα attributes θέλουμε όποτε όσο περισσότερα τόσο περισσότερες τροποποιήσεις θα μπορούμε να έχουμε.

```
function show_calendar($atts)
{
    $callW = $atts['calendarwidth'];
    $callH = $atts['calendarheight'];
    $callL = $atts['locale'];
```

Εικόνα 24 - Shortcode attributes

Το πιο περίπλοκο κομμάτι στα shortcodes που έχω φτιάξει είναι η επιλογή για συγκεκριμένες κατηγορίες. Θα πρέπει να δίνονται σαν attributes τα slug όλων των κατηγοριών αλλά και το νούμερο το κατηγοριών που δόθηκαν, κάτι που γίνεται αυτόματα με το shortcode generator και το μόνο που χρειάζεται να κάνει ο χρήστης είναι να επιλέξει τις κατηγορίες που θέλει να εμφανίζονται.

```
<style>
    .divCal {
        width: <?php echo $callW; ?>;
        height: <?php echo $callH; ?>;
    }
</style>
```

Εικόνα 25 - Χρήση των Attributes

Μέσα σε μια for για όσες κατηγορίες έχουν δοθεί βρίσκει το αντίστοιχο JSON αρχείο διαβάσει το περιεχόμενο και το προσθέτει σε μία μεταβλητή $data η οποία θα καταλήξει να περιέχει όλα τα events από τις κατηγορίες που επιλέχθηκαν. Όταν είναι να διαβάσει το full calendar ποια events να τυπώσει θα του δώσομε το αρχείο $data.

Τα shortcode δίνουν τη ελευθερία να δημιουργήσουμε οτιδήποτε θέλουμε σε μια μέθοδο και να το εμφανίσουμε πολύ ευκολά στο front-end του WordPress κάτι που με άλλο τρόπο θα ήταν πολύ περίπλοκο.

```
$countCats = $atts['countcats'];

$data = "";
for ($i = 1; $i <= $countCats; $i++) {
    ${"category" . $i} = $atts['cat' . $i];
    ${"url" . $i} = $url . "/" . ${"category" . $i} . ".json";
    ${"data" . $i} = file_get_contents(${"url" . $i});
    //echo ${"data" . $i} . "==========<br>";
    $data .= "" . ${"data" . $i} . "";
}
```

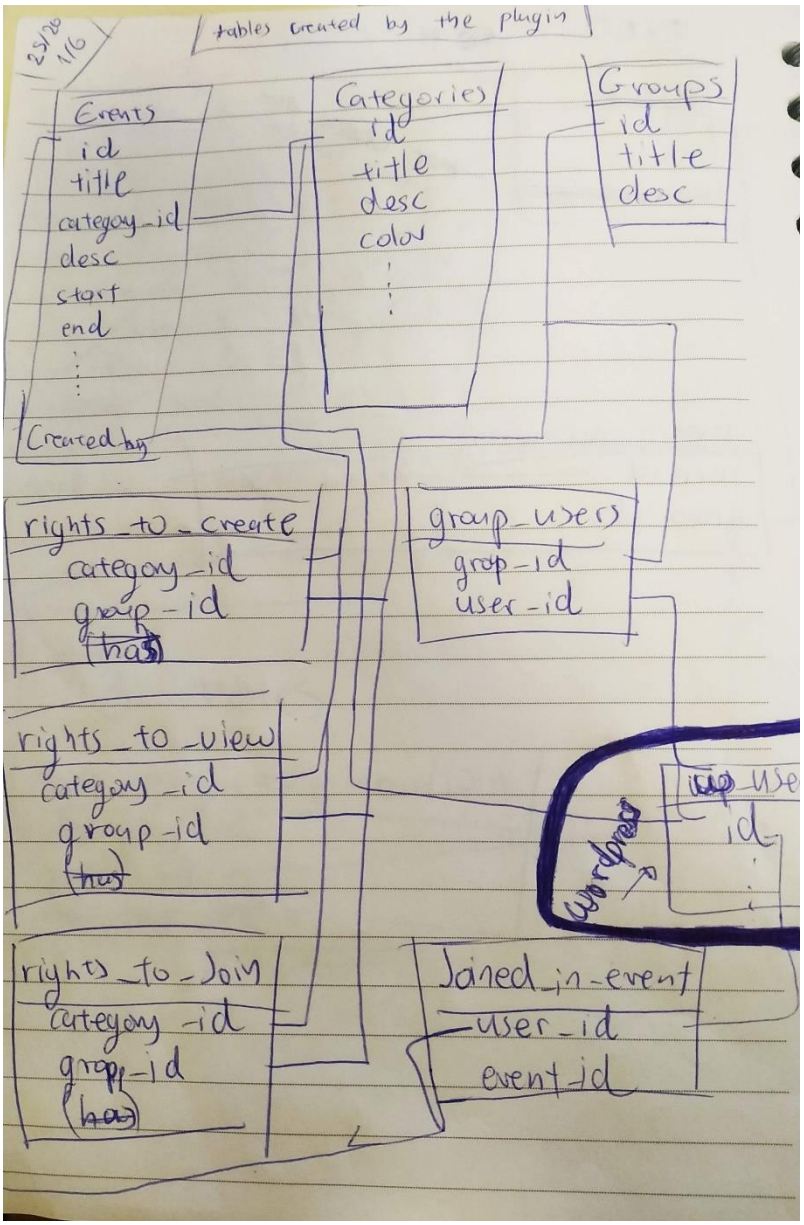
Εικόνα 26 - For loop για τη δημιουργία ενός αρχείου με περιεχόμενο σε JSON μορφή των κατηγοριών που έχουν δοθεί στο shortcode

5. Συζήτηση

5.1. Πορεία

1. Βάση και Menu

Όταν ξεκίνησα το plugin το πρώτο που έκανα ήταν η εμφάνιση του μενού μέσα στο WordPress Dashboard με τις 4 επιλογές το όποιο ήταν κάτι αρκετά εύκολο. Αμέσως μετά ήταν η βάση, από τη στιγμή που προσθέτω events μέσω του plugins ήθελα να αποθηκεύονται σε κάποιο διαφορετικό πίνακα. Το πρώτο σχέδιο που είχα κάνει για τη βάση ήταν αρκετά περίπλοκο με πολλούς πίνακες και όχι μόνο δύο που κατέληξα. Ήθελα να χρησιμοποιήσω τον ήδη υπάρχον πίνακα Users για ξένο κλειδί στο δικό μου πίνακα για να έχει κάθε event δημιουργό όπως και επιλογή για το ποιος θα παραβρεθεί σε ποιο event. Ο πίνακας με ξένο κλειδί από το πίνακα users δεν μπορούσε να δημιουργηθεί και ανάλογα με τα rights που θα είχε το άτομο το οποίο θα έκανε εγκατάσταση το plugin θα είχε θέμα με αυτό. Οπότε κατέληξα σε μια πιο απλή μορφή για τους πίνακες, ένας με κατηγορίες και ένας με events.



Εικόνα 27 - Παλιό πλάνο για τους πίνακες του πρόσθετου

2. ***ICS***

   Ένα σημαντικό κομμάτι των events είναι η δυνατότητα ανάγνωσης αρχείων iCalendar με κατάληξη .ics. Είχα δοκιμάσει διάφορα κομμάτια κώδικα που βρήκα σε PHP τελικά κατέληξα σε ένα που με μερικές αλλαγές έκανε αυτό που ήθελα. Σε πρώτη φάση όταν επέλεγα να κάνει import ένα αρχείο μου εμφάνιζε όλα τα στοιχεία του ics σε μορφή object και σιγά σιγά πρόσθεσα και κώδικα για να τα εισάγει στη βάση και να επιστρέφει μήνυμα επιτυχίας ή σφάλματος αν προσπαθήσει ο χρήστης να ανεβάσει ένα αρχείο που δεν έχει τη κατάληξη ics.

3. ***Προσθήκη***

   Προσπάθησα γενικά σε όλο το κώδικα να είναι οι φάκελοι αρκετά οργανωμένοι, έβαλα τις κλάσεις μόνες τους και καλούνται από τα αρχεία στο φάκελο config έτσι ώστε όλο το plugin να είναι κοντά σε μορφή MVC, όπου Model είναι τα αρχεία στο φάκελο Classes, View τα αρχεία στο φάκελο View τα οποία είναι αυτά που βλέπει ο χρήστης και Controller ότι είναι στο φάκελο config. Η προσθήκη γίνεται αρχικά μέσω του View και μέσω POST τα στοιχεία στέλνονται στο Config όπου εκεί δημιουργείται αντικείμενο τύπου Event, Category ή Cal και καλούνται οι μέθοδοι που χρειάζονται για να προστεθεί στη βάση.

4. ***Εμφάνιση***

   Στην εμφάνιση των events και των κατηγοριών είχα προσπαθήσει αρχικά να κάνω εγώ κάποιο πίνακα αλλά βρήκα τελικά μια JavaScript βιβλιοθήκη όπου τα εμφάνιζε πού καλύτερα και είχε επίσης τη δυνατότατά για αλλαγή στη σειρά με βάση κάποιο άλλο κριτήριό όπως η ημερομηνία αλλά το πιο σημαντικό έχει τη δυνατότητα αναζήτησης. Επίσης εμφανίζει τα events σε σελίδες ανά 10 ή και περισσότερα.

5. ***Διαγραφή και επεξεργασία***

   Η διαγραφή ήταν πολύ εύκολη στην υλοποίηση και για τα events και για της κατηγορίες ενώ η επεξεργασία όχι και τόσο. Η δυνατότητα επεξεργασίας ενός event ή μιας κατηγορίας ήταν το τελευταίο κομμάτι της εργασίας και είχα δοκιμάσει διάφορους τρόπους αλλά κατέληξα στη χρήση JSON .

5.2. Δυσκολίες

1. ***Paths***

   Ένα από το κομμάτια που με δυσκόλεψε πολύ ήταν ο ορισμός των paths. Σε πολλά σημεία του κώδικα έχω κομμάτια CSS μέσα στο View διότι δεν τα διάβασε από εξωτερικό αρχείο CSS. Κάτι που με βοήθησε αρκετά στο συγκεκριμένο πρόβλημα είναι η χρήση των WordPress Hook plugin_dir_path και plugin_dir_url όπου επιστρέφουν το path και το url μέχρι το φάκελο του plugin. Αν όλο το πρόγραμμα δεν ήταν ένα WordPress plugin άλλα ένα site σε php πολλά από τα θέματα που αντιμετώπισα με τα path δεν θα υπήρχαν.

2. ***JSON***

   Όταν προσπάθησα αρχικά να εμφανίσω events από τη βάση που είχα δημιουργήσει στο full calendar χρησιμοποιούσα την php εντολή json_encode αλλά δεν είχα καταφέρει κάτι. Είχα δοκιμάσει πολλά διαφορετικά Path και είχα δοκιμάσει να αφαιρέσω και τελείως το full calendar και να δοκιμάσω κάτι άλλο για την εμφάνιση. Τελικά από αποφάσισα να φτιάξω εγώ τα JSON βήμα-βήμα και με αυτό το τρόπο το full calendar μπορούσε να τα διαβάσει οπότε το εξέλιξα κι άλλο και έβαλα ένα JSON για κάθε κατηγορία.

3. ***Αναζήτηση***

   Ένα άλλο πρόβλημα που συνάντησα ήταν στην αναζήτηση. Αν προσπαθούσα να ψάξω για κάποιο πολύ συγκεκριμένο πρόβλημα για το plugin development κατά πάσα πιθανότητα το αποτέλεσμα να ήταν διάφορα έτοιμα plugin. Ένα μεγάλο κομμάτι της εργασίας ήταν να μπορέσω να “μεταφράσω” εντολές PHP σε WordPress Hooks.

4. ***Από Local σε Server***

   Όλη τη περίοδο που ασχολούμουνα με το plugin το δοκίμαζα σε local server μέσω του XAMPP, μόνο αφού το είχα έτοιμο το δοκίμασα σε Server και πρόσεξα ότι πολλά δεν δούλευαν. Εννοίται το πρόβλημα

ήταν πάλι τα paths κι ας μην είχα χρησιμοποιήσει πουθενά ABSOLUTE path. Το πρόβλημα λύθηκε με το WordPress Hook plugin_dir_path().

5.3. Μελλοντικές Επεκτάσεις

1. ***ICS***

   Κάποιες επεκτάσεις που έχω σκεφτεί σε σχέση με τα iCalendar αρχεία είναι να υπάρχει η δυνατότητα όχι μόνο για εισαγωγή τέτοιων αρχείων αλλά και η εξαγωγή, να μπορεί δηλαδή ο χρήστης να κάνει export τα events που έχει φτιάξει σε αυτή τη μορφή για να τα χρησιμοποιήσει και άλλου.

   Επίσης κατά την εισαγωγή αρχείων ics θα μπορούσε να υπάρχει και επιλογή για προσθήκη κατηγορίας εκείνη τη στιγμή, όπως και προβολή των αρχείων πριν εισαχθούν για να μπορεί ο χρήστης να επιλέξει ποια θα εισαχθούν.

2. ***Περισσότερα Settings***

   Το κομμάτι του shortcode δίνει αρκετή ελευθερία στο χρήστη για το τι θα φαίνεται αλλά όχι πολύ για τα χρώματα, μία ακόμα λειτουργεία θα ήταν να μπορεί να αλλάξει τα χρώματα από το calendar

3. ***Εμφάνιση***

   Ένα κομμάτι που δεν έχω ασχοληθεί ιδιαίτερα είναι η εμφάνιση. Όλες οι φόρμες είναι μεν κατανοητές αλλά πολύ απλές, θα μπορούσε μια μελλοντική εκδοχή να ήταν πιο όμορφη εμφανισιακά ίσως με τη χρήση Bootstrap.

4. ***FullCalendar***

   Το FullCalendar είναι μια βιβλιοθήκη που ακόμα και στη δωρεάν του έκδοση προσφέρει διάφορες λειτουργίες και πολλές από αυτές δεν υπάρχουν στο πρόσθετο. Μια μελλοντική επέκταση θα ήταν η καλύτερη εκμετάλλευση αυτής της βιβλιοθήκης

Βιβλιογραφία και Αναφορές

[1] Brad Williams, Justin Tadlock John and James Jacoby. *Professional WordPress Plugin Development: Edition 2*, 2020

[2] Aaron Brazell. *WordPress Bible: Edition 2*, 2011

[3] Thord and Daniel Hedengren. *Smashing WordPress Beyond the Blog*, 2012

[4] Jonathan Lazar. *User-centered Web development*, 2001

[5] Luke Welling and Laura Thomson. *PHP and MySQL Web development*, 2003

[6] Meet WordPress: WordPress is open source software you can use to create a beautiful website, blog, or app. [Online]. Available : https://wordpress.org/

[7] Web Development Roadmaps [Online]. Available : https://www.w3schools.com/whatis/default.asp

[8] The Top 100 Most Commonly Used WordPress Functions [Online]. Available : https://vegibit.com/the-top-100-most-commonly-used-wordpress-functions/

[9] Introduction to Plugin Development [Online]. Available https://developer.wordpress.org/plugins/intro/

[10] Creating A WordPress Plugin Is Easier Than You Think [Online]. Available : https://www.wpbeaverbuilder.com/creating-wordpress-plugin-easier-think/

[11] the most popular full-sized: JavaScript Calendar [Online]. Available https://fullcalendar.io/

[12] 7 Benefits of Using a Content Management System (CMS) [Online] Available: https://www.innovativearchitects.com/KnowledgeCenter/basic-IT-systems/CMS-Benefits.aspx

[13] 7 Advantages of Using a CMS to Run Your Site [Online] Available: https://www.enginess.io/insights/7-advantages-using-cms-run-site

[14] What is: Content Management System (CMS) [Online] Available: https://www.wpbeginner.com/glossary/content-management-system-cms/

[15] Choosing a CMS: What Does Open-Source Mean and Why Does it Matter? [Online] Available: https://www.websolutions.com/blog/choosing-a-cms-what-does-open-source-mean-and-why-does-it-matter/

[16] What Is WordPress? Explained for Beginners [Online] Available: https://kinsta.com/knowledgebase/what-is-wordpress/

[17] What does a front-end developer do? [Online] Available: https://www.careerexplorer.com/careers/front-end-developer/

Παράρτημα

wps_my_cal.php

```
<?php
/**
 * Plugin Name: MyCal
 * Description: Create your own calendar
 * Version: 1.0
 * Author: Marianna Krikou
 **/

include('classes/Cal.php');
include('classes/Category.php');
include('classes/Event.php');
include('classes/Json.php');
include("config/install.php");
include("config/unistall.php");
include("config/admin_menu.php");
include("config/calendar_page.php");
include("config/shortcode.php");
include("config/import-cal.php");
include("config/classes/category.php");
include("config/classes/event.php");
include("config/queries/fetch_event.php");
include("config/queries/fetch_category.php");
add_action('admin_menu', 'my_cal_options_panel');
register_activation_hook(__FILE__, 'create_plugin_database_table');
register_uninstall_hook(__FILE__, 'delete_plugin_database_table');
register_activation_hook(__FILE__, 'insert_page');
register_deactivation_hook(__FILE__, 'delete_page');
add_shortcode('view-calendar', 'show_calendar');
add_shortcode('view-calendarlist', 'show_calendar_list');
add_shortcode('view-calendar-choose-cat', 'show_calendar_choose');
add_shortcode('view-calendarlist-choose-cat', 'show_calendar_list_choose');
```

views/categories.php

```
<html>
<head>
   <meta charset="utf-8">
   <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="stylesheet" href="https://cdn.datatables.net/1.10.21/css/jquery.dataTables.min.css">
   <script src="https://code.jquery.com/jquery-3.5.1.js"></script>
   <script src="https://cdn.datatables.net/1.10.21/js/jquery.dataTables.min.js"></script>
   <style>
      .resnonsDiv {
         width: 40%;
         float: left;
         padding: 10px;
      }
      .row:after {
         content: "";
         display: table;
         clear: both;
      }
      #outer {
         text-align: center;
      }
      .inner {
         float: left;
         margin-right: 5px;
      }
      .tooltip {
         position: relative;
         display: inline-block;
      }
      .tooltip .tooltiptext {
         visibility: hidden;
         width: 120px;
```

```
    background-color: black;
    color: #fff;
    text-align: center;
    border-radius: 6px;
    padding: 5px 0;
    position: absolute;
    z-index: 1;
    bottom: 100%;
    left: 50%;
    margin-left: -60px;
    opacity: 0;
    transition: opacity 1s;
}
.tooltip:hover .tooltiptext {
    visibility: visible;
    opacity: 1;
}
.nostyle {
    border: none;
    padding: 0;
    background: none;
}
.nostyleOpen {
    border: none;
    padding: 0;
    background: none;
}
.popup-overlay {
    visibility: hidden;
    position: fixed;
}
.popup-overlay.active {
    visibility: visible;
```

```
    text-align: center;
    z-index: 1;
    padding-top: 100px;
    left: 0;
    top: 0;
    width: 100%;
    height: 100%;
    overflow: auto;
    background-color: rgb(0, 0, 0);
    background-color: rgba(0, 0, 0, 0.4);
}
.popup-content {
    visibility: hidden;
}
.popup-content.active {
    visibility: visible;
    position: relative;
    left: 20%;
    right: 10%;
    top: 0;
    width: 30%;
    height: 70%;
    background-color: #fefefe;
    margin: 10%;
    padding: 0;
    border: 1px solid #888;
}
.close {
    line-height: 12px;
    width: 18px;
    font-size: 8pt;
    font-family: tahoma;
    margin-top: 1px;
```

```
    margin-right: 2px;
    position: absolute;
    top: 0;
    right: 0;
  }
  .tab {
    overflow: hidden;
    border: 1px solid #ccc;
    background-color: #f1f1f1;
  }
  .tab button {
    background-color: inherit;
    float: left;
    border: none;
    outline: none;
    cursor: pointer;
    padding: 14px 16px;
    transition: 0.3s;
    font-size: 17px;
  }
  .tab button:hover {
    background-color: #ddd;
  }
  .tab button.active {
    background-color: #ccc;
  }
  .tabcontent {
    display: none;
    padding: 6px 12px;
    border: 1px solid #ccc;
    border-top: none;
  }
</style>
```

```
</head>
<body>
  <div class="tab">
    <button class="tablinks" id="defaultOpen" onclick="OpenTab(event, 'show-categories')">Categories Table</button>
    <button class="tablinks" onclick="OpenTab(event, 'add-category')">Create Category</button>
  </div>

  <?php
  $alerts_path = plugin_dir_path(__DIR__) . 'includes/alerts/';
  include $alerts_path . 'errors.php';
  include $alerts_path . 'corrects.php';
  include $alerts_path . 'warnings.php';
  ?>
  <div id="add-category" class="tabcontent">
    <form method="post">
      <h2>Add New Category</h2>
      Name<br>
      <input type="text" name="catTitle" required><br>
      Description<br>
      <input type="text" name="catDesc"><br>
      URL<br>
      <input type="text" name="catUrl"><br>
      Color<br>
      <div>
        <input type="color" name="catColor" value="#e66465">
      </div>
      <br><input type="submit" value="Add" name="addCat">
    </form>
  </div>
  <div id="show-categories" class="tabcontent">
    <table id="example" class="display">
      <thead>
        <tr>
```

```
        <th>Title</th>
        <th>Description</th>
        <th>Color</th>
      </tr>
    </thead>
    <tbody>
      <?php
      global $wpdb;
      $cat_table = $wpdb->prefix . 'wps_mc_categories';
      $query = "SELECT * FROM $cat_table";
      foreach ($wpdb->get_results($query) as $key => $row) {
        $id = $row->id;
        $title = $row->title;
        $description = $row->description;
        $color = $row->color;
        $image_path = plugin_dir_url(dirname(__FILE__)) . 'images/';
      ?><tr>
          <td>
            <div class="inner">
              <?php echo $title; ?>
            </div>
            <div class="inner">
              <div class="tooltip">
                <span class="tooltiptext">edit</span>
<button class="nostyleOpen" name="getCat" id=" <?php echo $id; ?>">
<img src="<?php echo $image_path . 'edit.png'; ?>" width="15" height="15" style="cursor: pointer;" name="editCat">
                </button>
              </div>
            </div>
            <form method="post">
       <div class="inner">
        <input type="hidden" name="deleteCatTitle" value=" <?php echo $title; ?>">
          <input type="hidden" name="deleteCatID" value=" <?php echo $id; ?>">
```

```
                    <div class="tooltip">
                  <span class="tooltiptext">delete</span>
<button class="nostyle" type="submit" name="deleteCat" onClick="return confirm('Are you sure you want to delete?')">
   <img src="<?php echo $image_path . 'delete.png'; ?>" width="15" height="15" style="cursor: pointer;" name="deleteCat">
                          </button>
                        </div>
                      </div>
                    </form>
                  </td>
                  <td><?php echo $description; ?></td>
                  <td style="background-color:<?php echo $color; ?>;"></td>
                </tr><?php
                  }
                    ?>
      </tbody>
      <tfoot>
        <tr>
          <th>Title</th>
          <th>Description</th>
          <th>Color</th>
        </tr>
      </tfoot>

    </table>

    <table cellspacing="0" cellpadding="0" border="0">
      <tbody>
        <tr>
          <td class="gutter">
            <div class="line number1 index0 alt2" style="display: none;">1</div>
          </td>
          <td class="code">
            <div class="container" style="display: none;">
```

```
                    <div class="line number1 index0 alt2" style="display: none;"> </div>
                </div>
            </td>
        </tr>
    </tbody>
</table>
</div>
<div class="popup-overlay">
    <div class="popup-content">
        <div class="modal-header">
            <button type="button" class="close" data-dismiss="modal">X</button>
        </div>
        <div class="modal-body" id="modal-body">
            <form method="post">
                <input type="hidden" name="updateCategoryId" id="updateId" value="">
                Name<br>
                <div id="category-title"></div><br>
                Description<br>
                <div id="category-desc"></div><br>
                URL<br>
                <div id="category-url"></div><br>
                Color<br>
                <div>
                    <div id="category-color"></div>
                </div>
                <br><input type="submit" value="Update" name="updateCat" />
                </table>
            </form>
        </div>
    </div>
</div>
</body>
<?php
```

```
$json_path = plugin_dir_path(__DIR__) . "includes/json/oneJSON/oneCat.json";
?>
<script>
   $(document).ready(function() {
      $('#example').DataTable({
         "pagingType": "full_numbers"
      });
      $(document).on('click', '.nostyleOpen', function() {
         var category_id = $(this).attr("id");
         console.log("test1");
         $.ajax({
            async: false,
            cache: false,
            timeout: 300,
            type: 'post',
            data: {
               "get_the_cat": category_id
            },
            success: function(response) {
            }
         });
         $(".popup-overlay, .popup-content").addClass("active");
         $.ajaxSetup({ cache: false });
         $.getJSON("../wp-content/plugins/wps-my-cal/includes/json/oneJSON/oneCat.json", function(json) {
            console.log(json);
 categoryTitle = '<input type="text" name="categoryTitleUp" value="' + json.title + '" required>'
            $("#category-title").html(categoryTitle);
categoryDescription = '<input type="text" name="categoryDescUp" value="' + json.description + '">'
            $("#category-desc").html(categoryDescription);
categoryUrl = '<input type="text" name="categoryUrlUp" value="' + json.url + '">'
            $("#category-url").html(categoryUrl);
categoryColor = '<input type="color" name="categoryColorUp" value=' + json.color + '>'
```

```
            $("#category-color").html(categoryColor);
            $("#updateId").val(category_id);
        });
    });
});
$(".close").on("click", function() {
    $(".popup-overlay, .popup-content").removeClass("active");
});
function OpenTab(evt, tabName) {
    var i, tabcontent, tablinks;
    tabcontent = document.getElementsByClassName("tabcontent");
    for (i = 0; i < tabcontent.length; i++) {
        tabcontent[i].style.display = "none";
    }
    tablinks = document.getElementsByClassName("tablinks");
    for (i = 0; i < tablinks.length; i++) {
        tablinks[i].className = tablinks[i].className.replace(" active", "");
    }
    document.getElementById(tabName).style.display = "block";
    evt.currentTarget.className += " active";
}
document.getElementById("defaultOpen").click();
</script>
</html>
```

view/events.php

```
<html>
<head>
   <meta charset="utf-8">
   <meta name="viewport" content="width=device-width, initial-scale=1">
   <link rel="stylesheet" href="//code.jquery.com/ui/1.12.1/themes/base/jquery-ui.css">
   <link rel="stylesheet" href="/resources/demos/style.css">
   <script src="https://code.jquery.com/jquery-1.12.4.js"></script>
   <script src="https://code.jquery.com/ui/1.12.1/jquery-ui.js"></script>
<link rel="stylesheet" href="https://cdn.datatables.net/1.10.21/css/jquery.dataTables.min.css">
   <script src="https://cdn.datatables.net/1.10.21/js/jquery.dataTables.min.js"></script>
   <script>
      $(function() {
         $("#datepicker1").datepicker();
      });
      $(function() {
         $("#datepicker2").datepicker();
      });
      $(function() {
         $("#datepicker3").datepicker();
      });
      $(function() {
         $("#datepicker4").datepicker();
      });
   </script>
   <style>
      #outer {
         text-align: center;

      .inner {
         float: left;
         margin-right: 5px;
      }
```

```
.tooltip {
    position: relative;
    display: inline-block;
}
.tooltip .tooltiptext {
    visibility: hidden;
    width: 120px;
    background-color: black;
    color: #fff;
    text-align: center;
    border-radius: 6px;
    padding: 5px 0;
    position: absolute;
    z-index: 1;
    bottom: 100%;
    left: 50%;
    margin-left: -60px;

    opacity: 0;
    transition: opacity 1s;
}
.tooltip:hover .tooltiptext {
    visibility: visible;
    opacity: 1;
}
.nostyle {
    border: none;
    padding: 0;
    background: none;
}
.nostyleOpen {
    border: none;
    padding: 0;
```

```
    background: none;
}
.popup-overlay {
    visibility: hidden;
    position: fixed;
}
.popup-overlay.active {
    visibility: visible;
    text-align: center;
    z-index: 1;
    padding-top: 100px;
    left: 0;
    top: 0;
    width: 100%;
    height: 100%;
    overflow: auto;
    background-color: rgb(0, 0, 0);
    background-color: rgba(0, 0, 0, 0.4);
}
.popup-content {
    visibility: hidden;
}
.popup-content.active {
    visibility: visible;
    position: relative;
    left: 20%;
    right: 10%;
    top: 0;
    width: 30%;
    height: 70%;
    background-color: #fefefe;
    margin: 10%;
    padding: 0;
```

```
    border: 1px solid #888;
}
.modal-header {
    text-align: center;
}
.close {
    width: 18px;
    font-size: 8pt;
    font-family: tahoma;
    margin-top: 1px;
    margin-right: 1px;
    position: absolute;
    top: 0;
    right: 0;
}
.tab {
    overflow: hidden;
    border: 1px solid #ccc;
    background-color: #f1f1f1;
}
.tab button {
    background-color: inherit;
    float: left;
    border: none;
    outline: none;
    cursor: pointer;
    padding: 14px 16px;
    transition: 0.3s;
    font-size: 17px;
    text-align: center;
}
.tab button:hover {
    background-color: #ddd;
```

```
    }
    .tab button.active {
        background-color: #ccc;
    }
    .tabcontent {
        display: none;
        padding: 6px 12px;
        border: 1px solid #ccc;
        border-top: none;
    }
  </style>

</head>

<body>
  <div class="tab">
<button class="tablinks" id="defaultOpen" onclick="OpenTab(event, 'show-events')">Events Table</button>
  <button class="tablinks" onclick="OpenTab(event, 'add-event')">Create Event</button>
 <button class="tablinks" onclick="OpenTab(event, 'add-ics')">Import .ics File</button>
  </div>
  <?php
  $alerts_path = plugin_dir_path(__DIR__) . 'includes/alerts/';
  include $alerts_path . 'errors.php';
  include $alerts_path . 'corrects.php';
  include $alerts_path . 'warnings.php';
  ?>
  <div class="row">
    <div>
      <div id="add-ics" class="tabcontent">
        <h3>Import .ics File</h3>
        <form method='POST' enctype='multipart/form-data'>
          <input type='file' name='icsFile'><br>
          <input type='submit' name='upload_btn' value='Upload'>
```

```
    </form>
  </div>
  <div id="add-event" class="tabcontent">
    <form method="post">
      <div>
        <h2>Add New Event</h2>
        Title<br>
        <input type="text" name="eventTitle" required><br>
        Description<br>
        <input type="text" name="eventDesc"><br>
        Start Date<br>
        <input type="text" id="datepicker1" name="startDate"><br>
        Start Time<br>
<input type="time" id="startTime" name="startTime" min="01:00" max="00:00" required><br>
        End Date<br>
        <input type="text" id="datepicker2" name="endDate"><br>
        End Time<br>
 <input type="time" id="endTime" name="endTime" min="01:00" max="00:00" required><br>
        Location<br>
        <input type="text" name="eventLocation"><br>
        URL<br>
        <input type="text" name="eventUrl"><br>
        Category<br>
        <select name="cat">
          <option selected="selected" value= "0">-</option>
          <?php
          global $wpdb;
          $categories_table = $wpdb->prefix . 'wps_mc_categories';
          $sql = "SELECT id,title FROM $categories_table";
          foreach ($wpdb->get_results($sql) as $key => $row) :
            $id = $row->id;
            $title = $row->title;
            echo "<option value=" . $id . ">" . $title . "</option>";
```

```
                endforeach;
                ?><br>
            </select>
        </div>
        <br><input type="submit" value="Add" name="addEvent">
    </form>
  </div>
  <br><br>
</div>
<div id="show-events" class="tabcontent">
  <table id="example" class="display">
    <thead>
      <tr>
        <th>Title</th>
        <th>Start</th>
        <th>End</th>
        <th>Location</th>
        <th>Category</th>
      </tr>
    </thead>
    <tbody>
      <?php
      global $wpdb;
      $events_table = $wpdb->prefix . 'wps_mc_events';
      $query = "SELECT * FROM $events_table";
      foreach ($wpdb->get_results($query) as $key => $row) {
        $id = $row->id;
        $start = $row->start;
        $end = $row->end;
        $title = $row->title;
        $location = $row->location;
        $description = $row->description;
        $cat_id = $row->category_id;
```

```
                $cat_table = $wpdb->prefix . 'wps_mc_categories';
                $queryCategory = "SELECT title from $cat_table WHERE id = $cat_id";
                $cat_title = $wpdb->get_var($queryCategory);
                $image_path = plugin_dir_url(dirname(__FILE__)) . 'images/';
            ?><tr>
                <td>
                  <div class="inner">
                    <?php echo $title; ?>
                  </div>
<input type="hidden" name="getEventID" value=" <?php echo $id; ?>">
                  <div class="inner">
                    <div class="tooltip">
                      <span class="tooltiptext">edit</span>
<button class="nostyleOpen" name="getEvent" id=" <?php echo $id; ?>">
<img src="<?php echo $image_path . 'edit.png'; ?>" width="15" height="15" style="cursor: pointer;" name="editEvent">
                      </button>
                    </div>
                  </div>
                  <form method="post">
                    <div class="inner">
<input type="hidden" name="deleteEventTitle" value=" <?php echo $title; ?>">
 <input type="hidden" name="deleteEventID" value=" <?php echo $id; ?>">
                      <div class="tooltip">
                        <span class="tooltiptext">delete</span>
 <button class="nostyle" type="submit" name="deleteEvent" onClick="return confirm('Are you sure you want to delete?')">
<img src="<?php echo $image_path . 'delete.png'; ?>" width="15" height="15" style="cursor: pointer;" name="deleteEvent">
                        </button>
                      </div>
                    </div>
                  </form>
                </td>
                <td><?php echo $start; ?></td>
                <td><?php echo $end; ?></td>
```

```
                <td><?php echo $location; ?></td>
                <td><?php echo $cat_title; ?></td>
              </tr><?php
                }
                  ?>
          </tbody>
          <tfoot>
            <tr>
              <th>Title</th>
              <th>Start</th>
              <th>End</th>
              <th>Location</th>
              <th>Category</th>
            </tr>
          </tfoot>
        </table>
        <table cellspacing="0" cellpadding="0" border="0">
          <tbody>
            <tr>
              <td class="gutter">
 <div class="line number1 index0 alt2" style="display: none;">1</div>
              </td>
              <td class="code">
                <div class="container" style="display: none;">
  <div class="line number1 index0 alt2" style="display: none;"> </div>
                </div>
              </td>
            </tr>
          </tbody>
        </table>
    </div>
</div>
<div class="popup-overlay">
```

```
<div class="popup-content">
  <div class="modal-header">
    <button type="button" class="close" data-dismiss="modal">X</button>
  </div>
  <div class="modal-body" id="modal-body">
    <form method="post">
      <input type="hidden" name="updateEventId" id="updateId" value="">
      <div id="left">
        Title<br>
        <div id="event-title"></div><br>
        Description<br>
        <div id="event-desc"></div><br>
        Start Date<br>
        <input type="text" id="datepicker3" name="startDateUp" value=""><br>
        Start Time<br>
        <div id="start-time"></div><br>
        End Date<br>
        <input type="text" id="datepicker4" name="endDateUp"><br>
        End Time<br>
        <div id="end-time"></div><br>
      </div>
      <div id="right">
        Location<br>
        <div id="event-location"></div><br>
        URL<br>
        <div id="event-url"></div><br>
        Category<br>
        <select name="catUp" id="selectCat">
          <option  value="0">-</option>
          <?php
          global $wpdb;
          $categories_table = $wpdb->prefix . 'wps_mc_categories';
          $sql = "SELECT id,title FROM $categories_table";
```

```
                foreach ($wpdb->get_results($sql) as $key => $row) :
                    $id = $row->id;
                    $title = $row->title;
                    echo "<option value=" . $id . ">" . $title . "</option>";
                endforeach;
                ?><br>
              </select>
            </div>
            <br><input type="submit" value="Update" name="updateEvent" />
          </form>
        </div>
      </div>
    </div>
  </body>
  <script>
    $(document).ready(function() {
      $('#example').DataTable({
        "pagingType": "full_numbers"
      });
      $(document).on('click', '.nostyleOpen', function() {
        var event_id = $(this).attr("id");
        $.ajax({
          async: false,
          cache: false,
          timeout: 300,
          type: 'post',
          data: {
            "get_the_event": event_id
          },
          success: function(response) {
          }
        });
        $(".popup-overlay, .popup-content").addClass("active");
```

```
        $.ajaxSetup({ cache: false });
$.getJSON("../wp-content/plugins/wps-my-cal/includes/json/oneJSON/oneEvent.json", function(json) {
eventTitle = '<input type="text" name="eventTitleUp" value="' + json.title + '" required>'
            $("#event-title").html(eventTitle);
eventDescription = '<input type="text" name="eventDescUp" value="' + json.description + '">'
            $("#event-desc").html(eventDescription);
eventLocation = '<input type="text" name="eventLocationUp" value="' + json.location + '">'
            $("#event-location").html(eventLocation);
eventUrl = '<input type="text" name="eventUrlUp" value=' + json.url + '>'
            $("#event-url").html(eventUrl);
startTime = '<input type="time" id="startTime" name="startTimeUp" value = "' + json.startTime + '" min="01:00" max="00:00">'
            $("#start-time").html(startTime);
endTime = '<input type="time" id="endTime" name="endTimeUp" value = "' + json.endTime + '" min="01:00" max="00:00">'
            $("#end-time").html(endTime);
            $("#datepicker3").val(json.startDate);
            $("#datepicker4").val(json.endDate);
            $("#selectCat").val(json.cat_id);
            $("#updateId").val(event_id);
        });
    });
});
$(".close").on("click", function() {
    $(".popup-overlay, .popup-content").removeClass("active");
});
function OpenTab(evt, tabName) {
    var i, tabcontent, tablinks;
    tabcontent = document.getElementsByClassName("tabcontent");
    for (i = 0; i < tabcontent.length; i++) {
        tabcontent[i].style.display = "none";
    }
    tablinks = document.getElementsByClassName("tablinks");
    for (i = 0; i < tablinks.length; i++) {
```

```
        tablinks[i].className = tablinks[i].className.replace(" active", "");
      }
      document.getElementById(tabName).style.display = "block";
      evt.currentTarget.className += " active";
    }
    document.getElementById("defaultOpen").click();
</script>
</html>
```

view/preview.php

```
<html>
<head>
  <style>
    .divCal {
      width: 60%;
      height: 60%;
    }
    .tab {
      overflow: hidden;
      border: 1px solid #ccc;
      background-color: #f1f1f1;
    }
    .tab button {
      background-color: inherit;
      float: left;
      border: none;
      outline: none;
      cursor: pointer;
      padding: 14px 16px;
      transition: 0.3s;
      font-size: 17px;
    }
    .tab button:hover {
      background-color: #ddd;
    }
    .tab button.active {
      background-color: #ccc;
    }
    .tabcontent {
      padding: 6px 12px;
      border: 1px solid #ccc;
      border-top: none;
```

```
    }
</style>
<?php
$packages_path = plugin_dir_url(__DIR__) . 'includes/fullCalendar/packages/';
wp_enqueue_style('fullcal-core', $packages_path . 'core/main.css');
wp_enqueue_style('fullcal-daygrid', $packages_path . 'daygrid/main.css');
wp_enqueue_style('fullcal-timegrid', $packages_path . 'timegrid/main.css');
wp_enqueue_style('fullcal-interaction', $packages_path . 'interaction/main.css');
//js
wp_enqueue_script('fullcal-core', $packages_path . 'core/main.js');
wp_enqueue_script('fullcal-daygrid', $packages_path . 'daygrid/main.js');
wp_enqueue_script('fullcal-timegrid', $packages_path . 'timegrid/main.js');
wp_enqueue_script('fullcal-interaction', $packages_path . 'interaction/main.js');
$url = plugin_dir_path(__DIR__) . "includes/json/allEvent.json";
$data = file_get_contents($url);
?>
<script>
    document.addEventListener('DOMContentLoaded', function() {
        var calendarEl = document.getElementById('calendar');
        var calendar = new FullCalendar.Calendar(calendarEl, {
            eventClick: function(info) {
                var eventObj = info.event;
                if (eventObj.url) {
                    window.open(eventObj.url);
                    info.jsEvent.preventDefault();
                } else {
                    alert(eventObj.title + '.\n' + eventObj.start);
                }
            },
            plugins: ['dayGrid', 'timeGrid', 'interaction'],
            timeZone: 'local',
            eventLimit: true,
            nowIndicator: true,
```

```
            locale: 'el',
            header: {
               left: 'dayGridMonth,timeGridWeek,timeGridDay custom1',
               center: '',
               right: 'title',
            },
            footer: {
               left: 'today',
               center: '',
               right: 'custom2 prevYear,prev,next,nextYear'
            },
            events: <?php echo "[" . $data  . "]"; ?>,
            eventTimeFormat: {
               hour: '2-digit',
               minute: '2-digit',
               meridiem: false
            },
         });
         calendar.render();
      });
   </script>
   <script>
      document.addEventListener('DOMContentLoaded', function() {
         var calendarEl = document.getElementById('calendarlist');
         var calendar2 = new FullCalendar.Calendar(calendarEl, {
            plugins: ['list'],
            timeZone: 'UTC',
            defaultView: 'listWeek',
views: { listDay: { buttonText: 'day' },listWeek: { buttonText: 'week' }, listMonth: { buttonText: 'month' },listYear: {buttonText:
'year'}, },
            header: {
               left: 'title',
               center: '',
```

```
            right: "'listDay,listWeek,listMonth,listYear'"
          },
          events: <?php echo "[" . $data . "]"; ?>,
          eventTimeFormat: {
            hour: '2-digit',
            minute: '2-digit',
            meridiem: false
          }
        });
        calendar2.render();
      });
  </script>
</head>
<body>
  <div class="tab">
 <button class="tablinks" id="defaultOpen" onclick="OpenTab(event, 'full-cal')"> Full Calendar Example</button>
<button class="tablinks" onclick="OpenTab(event, 'full-list')">Full Calendar List Example</button>
  </div>
  <div id="full-cal" class="tabcontent">
    <div class="divCal" id="calendar"></div>
  </div>
  <div id="full-list" class="tabcontent">
    <div class="divCal" id="calendarlist"></div>
  </div>
</body>
<script>
    function OpenTab(evt, tabName) {
      var i, tabcontent, tablinks;
      tabcontent = document.getElementsByClassName("tabcontent");
      for (i = 0; i < tabcontent.length; i++) {
        tabcontent[i].style.display = "none";
      }
      tablinks = document.getElementsByClassName("tablinks");
```

```
        for (i = 0; i < tablinks.length; i++) {
            tablinks[i].className = tablinks[i].className.replace(" active", "");
        }
        document.getElementById(tabName).style.display = "block";
        evt.currentTarget.className += " active";
    }
    document.getElementById("defaultOpen").click();
</script>
</html>
```

view/shortcode.php

```
<html>
<head>
   <script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.1/jquery.min.js"></script>
</head>
<?php
global $wpdb;
$categories_table = $wpdb->prefix . 'wps_mc_categories';
$sql = "SELECT title,slug FROM $categories_table";
?>
<style>
   .shown {
      visibility: visible;
   }

   .hidden {
      visibility: hidden;
   }
</style>
<body>
   <div>
      <h2>Calendar with all events</h2>
      <label style="font-size: 15px;" id="fullCalendar">[view-calendar calendarwidth="70%" calendarheight="70%" locale="en-gb"]</label>
      <button onclick="copyToClipboard('#fullCalendar')">Copy</button>
      </form>
   </div>
<br>
   <div>
      <h2>List with all events</h2>
      <label style="font-size: 15px;" id="fullCalendar2">[view-calendarlist calendarwidth="100%" calendarheight="100%"]</label>
      <button onclick="copyToClipboard('#fullCalendar2')">Copy</button>
      </form>
```

```
    </div>
    <br>
    <div>
        <h2>Create Custom</h2>
        <label for="calendarType">Choose Type: </label><br>
<input onclick="hideDiv()" type="radio" name="calendar_type" value="calendar" checked="checked">Calendar</input><br>
 <input onclick="showDiv(); updateCheckedView()" type="radio" name="calendar_type" value="calendarlist">List</input><br>
        <br>
        <div id="myDIV">
            Choose Views:<br>
<input type="checkbox" onclick="updateCheckedView()" id="views" name="views" value="D">Day
<input type="checkbox" onclick="updateCheckedView()" id="views" name="views" value="W">Week
<input type="checkbox" onclick="updateCheckedView()" id="views" name="views" value="M">Month
 <input type="checkbox" onclick="updateCheckedView()" id="views" name="views" value="Y">Year
        </div>
        <br>
        Categories shown:<br>
<input onclick="hideDiv2()" type="radio" name="categories_option" value="0" checked="checked">All Categories</input><br>
<input onclick="showDiv2(); updateChecked()" type="radio" name="categories_option" value="choose-cat">Choose
Categories</input><br>
        <br>
        <div id="allCats">
            Choose Event Categories:<br>
            <?php
            foreach ($wpdb->get_results($sql) as $row) :
                $title = $row->title;
                $slug = $row->slug;
                echo "<input type='checkbox' onclick='updateChecked()' class='cats' name='cats' id='cats' value=" . $slug .
">$title<br>";
            endforeach;
            ?>
        </div>
        <br>
        Choose Language:<br>
```

```
    <select id="locale" name="locale">
      <option value="el" selected>el</option>
      <option value="en-gb">en-gb</option>
    </select>
    <br><br>
    Height:<br>
    <input type="range" min="1" max="100" value="50" class="slider" id="myHeight">
    <br>
    Width:<br>
    <input type="range" min="1" max="100" value="50" class="slider2" id="myWidth">
    <br>
    <label style="font-size: 15px;" id="customShortCode">[view-<span id="view_type">calendar</span><span
id="choose_cat_opt"></span> calendarwidth="<span id="printHeight">50</span>%" calendarheight="<span
id="printWidth">50</span>%"<span id="views_out"></span><span id="show_cats"></span> locale="<span
id="localeOut">el</span>"]</label>

    <button onclick="copyToClipboard('#customShortCode')">Copy</button>
    <br>
  </div>
</body>
<script>
  function copyToClipboard(element) {
    var $temp = $("<input>");
    $("body").append($temp);
    $temp.val($(element).text()).select();
    document.execCommand("copy");
    $temp.remove();
  }
  var slider = document.getElementById("myHeight");
  var output = document.getElementById("printHeight");
  output.innerHTML = slider.value;
  slider.oninput = function() {
    output.innerHTML = this.value;
  }
```

```
var slider2 = document.getElementById("myWidth");
var output2 = document.getElementById("printWidth");
output2.innerHTML = slider2.value;
slider2.oninput = function() {
  output2.innerHTML = this.value;
}
var locale = document.getElementById("locale");
localeOut.innerHTML = locale.value;
locale.oninput = function() {
  localeOut.innerHTML = this.value;
}
$('document').ready(
  function() {
    $('input:radio[name="calendar_type"]').click(
      function() {
        $("#view_type").html(event.target.value)
      }
    );
  }
);
var x = document.getElementById("myDIV");
x.style.display = "none";
function showDiv() {
  var x = document.getElementById("myDIV");
  x.style.display = "block";
  $("#viewOpt").removeClass("hidden");
}
function hideDiv() {
  var x = document.getElementById("myDIV");
  x.style.display = "none";
  $("#viewOpt").addClass("hidden");
  $("#views_out").html("");
}
```

```
var y = document.getElementById("allCats");
y.style.display = "none";
function showDiv2() {
    var y = document.getElementById("allCats");
    y.style.display = "block";
    $("#choose_cat_opt").html("-choose-cat");
}
function hideDiv2() {
    var y = document.getElementById("allCats");
    y.style.display = "none";
    $("#choose_cat_opt").html("");
    $("#show_cats").html("");
}
function updateChecked() {
    var allCats = $('#cats:checked');
    var countOfCats = allCats.length;
    var cat = "";
    var counter = 0;
    var outputString = "";
    var outputCount = "";
    var finalOutput = "";
    for (var i = 0; i < countOfCats; i++) {
        cat = allCats[i].value;
        counter++;
        outputString += 'cat' + counter + ' = "' + cat + '" ';
        outputCount = ' countcats = "' + counter + '"';
        finalOutput = outputCount + ' ' + outputString;
    }
    $("#show_cats").html(finalOutput);
}
function updateCheckedView() {
    var allViews = $('#views:checked');
    var countViews = allViews.length;
```

```
    var view = "";
    var counter = 0;
    var outputString = "";
    var finalOutput = "";
    for (var i = 0; i < countViews; i++) {
      view = allViews[i].value;
      outputString += view;
      finalOutput = ' view = "' + outputString + '"';
    }
    $("#views_out").html(finalOutput);
  }
</script>
</html>
```

classes/Category.php

```
<?php
class Category {
   function __construct(){    }
   var $catTitle;
   var $catDesc;
   var $catColor;
   var $catUrl;
   var $catSlugl;
   public function addCategory($catTitle, $catSlug, $catDesc, $catUrl, $catColor)    {
      global $wpdb;
      $categories_table = $wpdb->prefix . 'wps_mc_categories';
      $catData = array(
         'title' => $catTitle,
         'slug' => $catSlug,
         'description' => $catDesc,
         'url' => $catUrl,
         'color' => $catColor );
      $wpdb->insert($categories_table, $catData);
      $target_Path = plugin_dir_path(__DIR__) . "includes/json/categorized/";
  $file = fopen($target_Path . "/" . $catTitle . ".json", 'w') or die("can't open file");
      fclose($file);    }
   public function updateCategory($catId, $catTitle, $catSlug, $catDesc, $catUrl, $catColor)
   {
      global $wpdb;
      $categories_table = $wpdb->prefix . 'wps_mc_categories';
      $eventData = array(
         'title' => $catTitle,
         'slug' => $catSlug,
         'description' => $catDesc,
         'url' => $catUrl,
         'color' => $catColor
      );
```

```
        $where = array(
            'id' => $catId
        );
        $wpdb->update($categories_table, $eventData, $where);
    }
    public function deleteCategory($catId)    {
        global $wpdb;
        $categories_table = $wpdb->prefix . 'wps_mc_categories';

        $where = array(
            'id' => $catId
        );
        $wpdb->delete($categories_table, $where);
    }
}
```

classes/Events.php

```
<?php
class Event
{
   function __construct()
   {
   }
   var $eventTitle;
   var $eventDesc;
   var $start;
   var $end;
   var $eventLocation;
   var $eventCat;
   var $url;
   var $uid;
   var $status;
   var $created;
   var $updated;
   var $isReccurent;
   var $eventId;
   public function addEvent($eventTitle, $eventDesc, $start, $end, $eventLocation, $eventCat, $url)
   {
      global $wpdb;
      $events_table = $wpdb->prefix . 'wps_mc_events';
      $eventData = array(
         'title' => $eventTitle,
         'description' => $eventDesc,
         'start' => $start,
         'end' => $end,
         'location' => $eventLocation,
         'created' => current_time('mysql', +3), //gmt not working
         'category_id' => $eventCat,
         'url' => $url
```

```
        );
        $wpdb->insert($events_table, $eventData);
    }
    public function addEventWithoutCat($eventTitle, $eventDesc, $start, $end, $eventLocation, $url)
    {
        global $wpdb;
        $events_table = $wpdb->prefix . 'wps_mc_events';
        $eventData = array(
            'title' => $eventTitle,
            'description' => $eventDesc,
            'start' => $start,
            'end' => $end,
            'location' => $eventLocation,
            'created' => current_time('mysql', +3),
            'url' => $url
        );
        $wpdb->insert($events_table, $eventData);
    }
    public function addEventFromICS($eventTitle, $uid,  $start, $end, $eventDesc, $eventLocation, $status, $created, $updated,
$isReccurent)    {
        global $wpdb;
        $table = $wpdb->prefix . 'wps_mc_events';
        $dataICS = array(
            'title' => $eventTitle,
            'uid' => $uid,
            'start' => $start,
            'end' => $end,
            'description' => $eventDesc,
            'location' => $eventLocation,
            'status' => $status,
            'created' => $created,
            'updated' => $updated,
            'isReccurent' => $isReccurent
```

```
        );
        $wpdb->insert($table, $dataICS);
    }
    public function addEventFromICSwithCat($eventTitle, $uid,  $start, $end, $eventDesc, $eventLocation, $status, $created,
$updated, $isReccurent)    {
        global $wpdb;
        $table = $wpdb->prefix . 'wps_mc_events';
        $dataICS = array(
            'title' => $eventTitle,
            'uid' => $uid,
            'start' => $start,
            'end' => $end,
            'description' => $eventDesc,
            'location' => $eventLocation,
            'status' => $status,
            'created' => $created,
            'updated' => $updated,
            'isReccurent' => $isReccurent
        );
        $wpdb->insert($table, $dataICS);
    }
    public function updateEvent($eventId, $eventTitle, $eventDesc, $start, $end, $eventLocation, $eventCat, $url)    {
        global $wpdb;
        $events_table = $wpdb->prefix . 'wps_mc_events';
        $eventData = array(
            'title' => $eventTitle,
            'description' => $eventDesc,
            'start' => $start,
            'end' => $end,
            'location' => $eventLocation,
            'updated' => current_time('mysql', +3), //gmt not working
            'category_id' => $eventCat,
            'url' => $url
```

```
    );
    $where = array(
        'id' => $eventId
    );
    $wpdb->update($events_table, $eventData, $where);
}
public function updateEventWithouCat($eventId, $eventTitle, $eventDesc, $start, $end, $eventLocation, $url)    {
    global $wpdb;
    $events_table = $wpdb->prefix . 'wps_mc_events';
    $eventData = array(
        'title' => $eventTitle,
        'description' => $eventDesc,
        'start' => $start,
        'end' => $end,
        'location' => $eventLocation,
        'updated' => current_time('mysql', +3), //gmt not working
        'url' => $url
    );
    $where = array(
        'id' => $eventId
    );
    $wpdb->update($events_table, $eventData, $where);
}
public function deleteEvent($eventId)    {
    global $wpdb;
    $events_table = $wpdb->prefix . 'wps_mc_events';

    $where = array(
        'id' => $eventId
    );
    $wpdb->delete($events_table, $where);
}
public function updateDeletedCatEvents($cat_id)    {
```

```
        global $wpdb;
        $events_table = $wpdb->prefix . 'wps_mc_events';
        $eventData = array(
            'category_id' => null
        );
        $where = array(
            'category_id' => $cat_id
        );
        $wpdb->update($events_table, $eventData, $where);
    }
}
```

classes/Json.php

```
<?php
class Json
{
   function __construct()
   {
   }
   var $cat_name;
   public function updateAllJSON()
   {
      global $wpdb;
      $events_table = $wpdb->prefix . 'wps_mc_events';
      $query = "SELECT * FROM $events_table ORDER BY id";
      $target_Path = plugin_dir_path(__DIR__) . "includes/json/";
      $fp = fopen($target_Path . 'allEvent.json', 'w');
      foreach ($wpdb->get_results($query) as $row) :
         $start = $row->start;
         $end = $row->end;
         $title = $row->title;
         $location = $row->location;
         $description = $row->description;
         $category_id = $row->category_id;
         $url = $row->url;
         $cat_table = $wpdb->prefix . 'wps_mc_categories';
         $query2 = "SELECT color FROM $cat_table WHERE id=$category_id";
         $cat_color = $wpdb->get_var($query2);
         if (empty($url)) {
            $query3 = "SELECT url FROM $cat_table WHERE id=$category_id";
            $url = $wpdb->get_var($query3);
         }
         $event = '
            {
            "start": "' . $start . '",
```

```
            "end": "' . $end . '",
            "title": "' . $title . '",
            "location": "' . $location . '",
            "description": "' . $description . '",
            "color": "' . $cat_color . '",
            "url": "' . $url . '"
            },
    ';
        fwrite($fp, $event);
    endforeach;
    fclose($fp);
}
public function updateAllCategoriesJSON()
{
    $target_Path = plugin_dir_path(__DIR__) . "includes/json/categorized/*";
    $files = glob($target_Path); // get all file names
    foreach ($files as $file) {
        if (is_file($file))
            unlink($file); // delete file
    }
    global $wpdb;
    $categories_table = $wpdb->prefix . 'wps_mc_categories';
    $get_categories = "SELECT * FROM $categories_table";
    foreach ($wpdb->get_results($get_categories) as $row) :
        $id = $row->id;
        $title = $row->title;
        $color = $row->color;
        $slug = $row->slug;
        $catUrl = $row->url;
        $target_Path2 = plugin_dir_path(__DIR__) . "includes/json/categorized";
      $fp = fopen($target_Path2 . "/" . $slug . ".json", 'w') or die("can't open file");
        $events_table = $wpdb->prefix . 'wps_mc_events';
        $getEvent = "SELECT * FROM $events_table WHERE category_id = $id";
```

```
    foreach ($wpdb->get_results($getEvent) as $row) :
        $start = $row->start;
        $end = $row->end;
        $title = $row->title;
        $location = $row->location;
        $description = $row->description;
        $url = $row->url;
        if (empty($url)) {
            $url = $catUrl;
        }
        $event = '
            {
            "start": "' . $start . '",
            "end": "' . $end . '",
            "title": "' . $title . '",
            "location": "' . $location . '",
            "description": "' . $description . '",
            "color": "' . $color . '",
            "url": "' . $url . '"
            },
                ';
        fwrite($fp, $event);
    endforeach;
    fclose($fp);
endforeach;
$fp2 = fopen(plugin_dir_path(__DIR__) . "includes/json/categorized/uncategorized.json", 'w') or die("can't open file");
$events_table = $wpdb->prefix . 'wps_mc_events';
$getEvent = "SELECT * FROM $events_table WHERE category_id is NULL";
foreach ($wpdb->get_results($getEvent) as $row) :
    $start = $row->start;
    $end = $row->end;
    $title = $row->title;
    $location = $row->location;
```

```
        $description = $row->description;
        $url = $row->url;
        $event = '
          {
          "start": "' . $start . '",
          "end": "' . $end . '",
          "title": "' . $title . '",
          "location": "' . $location . '",
          "description": "' . $description . '",
          "url": "' . $url . '"
          },
            ';
        fwrite($fp2, $event);
      endforeach;
      fclose($fp2);
    }
    public function createOneEventJSON($event_id)
    {
      global $wpdb;
      $event_table = $wpdb->prefix . 'wps_mc_events';
$fp = fopen(plugin_dir_path(__DIR__) . "includes/json/oneJSON/oneEvent.json", 'w')or die("can't open file");
      $get_the_event = "SELECT * FROM $event_table WHERE `id` = '" . $event_id . "'";
      $row = $wpdb->get_row($get_the_event);
      $start = $row->start;
      $end = $row->end;
      $title = $row->title;
      $location = $row->location;
      $description = $row->description;
      $url = $row->url;
      $cat_id = $row->category_id;
      $start = preg_split("/[\s,]+/", $start);
      $startDate = $start[0];
      $startTime = $start[1];
```

```
    $end = preg_split("/[\s,]+/", $end);
    $endDate = $end[0];
    $endTime = $end[1];
    $startDate = date("m/d/Y", strtotime($startDate));
    $endDate = date("m/d/Y", strtotime($endDate));
    $event = '
        {
        "startDate": "' . $startDate . '",
        "startTime": "' . $startTime . '",
        "endDate": "' . $endDate . '",
        "endTime": "' . $endTime . '",
        "title": "' . $title . '",
        "location": "' . $location . '",
        "description": "' . $description . '",
        "url": "' . $url . '",
        "cat_id": "' . $cat_id . '"
        }
    ';
    fwrite($fp, $event);
    fclose($fp);
}
public function createOneCatJSON($cat_id)
{
    global $wpdb;
    $event_table = $wpdb->prefix . 'wps_mc_categories';
    $target_Path = plugin_dir_path(__DIR__) . "includes/json/oneJSON";
    $fp = fopen($target_Path . "/oneCat.json", 'w');
    $get_the_cat = "SELECT * FROM $event_table WHERE `id` = '" . $cat_id . "'";
    $row = $wpdb->get_row($get_the_cat);
    $title = $row->title;
    $slug = $row->slug;
    $description = $row->description;
    $color = $row->color;
```

```
        $url = $row->url;
        $category = '
            {
            "title": "' . $title . '",
            "slug": "' . $slug . '",
            "description": "' . $description . '",
            "color": "' . $color . '",
            "url": "' . $url . '"
            }
        ';
        fwrite($fp, $category);
        fclose($fp);
    }
}
```

config/admin_menu.php

```
<?php
function my_cal_options_panel()
{
   add_menu_page(
      'My Calender page title',
      'My Calendar',
      'manage_options',
      'cal-options',
      'wps_cal_func',
      plugin_dir_url(__FILE__) . '../images/cal-icon-3.png'
   );
   add_submenu_page(
      'cal-options',
      'url page title',
      'Events',
      'manage_options',
      plugin_dir_path(__DIR__) . 'view/events.php',
      null
   );
   add_submenu_page(
      'cal-options',
      'url page title',
      'Categories',
      'manage_options',
      plugin_dir_path(__DIR__) . 'view/categories.php',
      null
   );
   add_submenu_page(
      'cal-options',
      'url page title',
      'Shortcodes',
      'manage_options',
```

```
        plugin_dir_path(__DIR__) . 'view/shortcodes.php',
        null
    );
    add_submenu_page(
        'cal-options',
        'url page title',
        'Example preview',
        'manage_options',
        plugin_dir_path(__DIR__) . 'view/preview.php',
        null
    );
}
function wps_cal_func()
{
    echo '<h2> Χρήση του Πρόσθετου </h2><br>';
}
```

config/calendar_page.php

```
<?php
function insert_page()
{
   $my_post = array(
      'post_title'    => 'My Calendar',
      'post_content'  => '[view-calendar calendarwidth="70%" calendarheight="70%" locale="en-gb"]',
      'post_status'   => 'publish',
      'post_author'   => get_current_user_id(),
      'post_type'     => 'page',
   );
   $create_mycal_page = wp_insert_post($my_post, false);
   update_option('mycalpage', $create_mycal_page);
}
function delete_page()
{
   $page_id = get_option('mycalpage');
   wp_delete_post($page_id);
}
```

config/import-cal.php

```
<?php
  global $wpdb;
  $events_table = $wpdb->prefix . 'wps_mc_events';
  $errors = array();
  $corrects = array();
  $warnings = array();
  if (isset($_POST['upload_btn'])) {
    $target_Path = plugin_dir_path(__DIR__) . "includes/ics-temp/";
    $target_Path = $target_Path . basename($_FILES['icsFile']['name']);
    $icsFileType = strtolower(pathinfo($target_Path, PATHINFO_EXTENSION));
    if ($_FILES["icsFile"]["error"] != 0) {
      array_push($errors, "No file selected.");       }
    if (!empty($icsFileType) && $icsFileType != "ics") {
      array_push($errors, "Sorry, only .ics files are allowed.");       }
if (count($errors) == 0 && move_uploaded_file($_FILES['icsFile']['tmp_name'], $target_Path)){
      $file = __DIR__ . "/../includes/ics-temp/" . $_FILES['icsFile']['name'];
      $iCal = new iCal($file);
      unlink($file);
      $events = $iCal->eventsByDate();
      $countEvents = 0;
      $countDuplicateUID = 0;
      foreach ($events as $date => $events) {
        foreach ($events as $event) {
          $event_obj = new Event();
          $title = $event->title();
          $uid = $event->uid();
          $start = $event->timeStart();
          $end = $event->timeEnd();
          $description = $event->description();
          $isReccurent = $event->isRecurrent();
          $location =  $event->location();
          $status = $event->status();
```

```
            $created = $event->created();
            $updated = $event->updated();
            $checkSameUid = $wpdb->get_var("SELECT COUNT(*) FROM $events_table WHERE uid = '$uid'");
            if ($checkSameUid > 0) {
               $countDuplicateUID++;                }

            $event_obj->addEventFromICS($title, $uid, $start, $end, $description,  $location, $status, $created, $updated,
$isReccurent);
            $countEvents++;             }         }
        array_push($corrects, $countEvents . " event(s) added");
        if ($countDuplicateUID > 0) {
           array_push($warnings, $countDuplicateUID . " event(s) duplicated");
        }
     }
  }
config/install.php
<?php
function create_plugin_database_table()
{
   global $wpdb;
   $categories_table = $wpdb->prefix . 'wps_mc_categories';
   $events_table = $wpdb->prefix . 'wps_mc_events';
   $charset_collate = $wpdb->get_charset_collate();
   $create_categories = "CREATE TABLE $categories_table (
            id int(11) NOT NULL auto_increment,
            title varchar(256) NOT NULL,
            slug varchar(256) NOT NULL,
            description text,
            color text NOT NULL,
            url varchar(255),
            PRIMARY KEY (id)
        )$charset_collate;";
   $create_events = "CREATE TABLE $events_table (
```

```
        id int(11) NOT NULL auto_increment,
        uid varchar(255),
        title varchar(256) NOT NULL,
        start datetime NOT NULL,
        end datetime NOT NULL,
        description text,
        location text,
        status varchar(255),
        isReccurent BOOLEAN,
        created datetime,
        updated datetime,
        category_id int(11),
        url varchar(255),
        PRIMARY KEY (id),
        FOREIGN KEY (category_id) REFERENCES  $categories_table(id)
    )$charset_collate;";
  require_once(ABSPATH . 'wp-admin/includes/upgrade.php');
  dbDelta($create_categories);
  dbDelta($create_events);
  add_option('test_db_version', $test_db_version);
}
```

config/unistall.php

```
<?php
function delete_plugin_database_table()
{
  global $wpdb;
  $categories_table = $wpdb->prefix . 'wps_mc_categories';
  $events_table = $wpdb->prefix . 'wps_mc_events';
  $delete_events = "DROP TABLE IF EXISTS $events_table";
  $detele_categories = "DROP TABLE IF EXISTS $categories_table";
  $wpdb->query($delete_events);
  $wpdb->query($detele_categories);
  delete_option("test_db_version");
```

config/shortcode.php

```
<?php
$packages_path = plugin_dir_url(__DIR__) . 'includes/fullCalendar/packages/';
wp_enqueue_style('fullcal-core', $packages_path . 'core/main.css');
wp_enqueue_style('fullcal-daygrid', $packages_path . 'daygrid/main.css');
wp_enqueue_style('fullcal-timegrid', $packages_path . 'timegrid/main.css');
wp_enqueue_style('fullcal-interaction', $packages_path . 'interaction/main.css');
wp_enqueue_style('fullcal-list', $packages_path . 'list/main.css');
wp_enqueue_script('fullcal-core', $packages_path . 'core/main.js');
wp_enqueue_script('fullcal-daygrid', $packages_path . 'daygrid/main.js');
wp_enqueue_script('fullcal-timegrid', $packages_path . 'timegrid/main.js');
wp_enqueue_script('fullcal-interaction', $packages_path . 'interaction/main.js');
wp_enqueue_script('fullcal-list', $packages_path . 'list/main.js');
function show_calendar($atts)
{
   $callW = $atts['calendarwidth'];
   $callH = $atts['calendarheight'];
   $callL = $atts['locale'];
   $url = plugin_dir_url(__DIR__) . "includes/json/allEvent.json";
   $data = file_get_contents($url);
?>
   <script>
      document.addEventListener('DOMContentLoaded', function() {
         var calendarEl = document.getElementById('calendar');
         var calendar = new FullCalendar.Calendar(calendarEl, {
            plugins: ['dayGrid', 'timeGrid', 'interaction'],
            timeZone: 'local',
            eventLimit: true,
            nowIndicator: true,
            displayEventTime: false,
            locale: '<?php echo $callL ?>',
            header: {
               left: 'dayGridMonth,timeGridWeek,timeGridDay custom1',
```

```
            center: '',
            right: 'title',
          },
          footer: {
            left: 'today',
            center: '',
            right: 'custom2 prevYear,prev,next,nextYear'
          },
          events: <?php echo "[" . $data . "]"; ?>,
          eventTimeFormat: {
            hour: '2-digit',
            minute: '2-digit',
            meridiem: false
          }
        });
        calendar.render();
      });
    </script>
    <style>
      .divCal {
        width: <?php echo $callW; ?>;
        height: <?php echo $callH; ?>;
      }
    </style>
<?php
    return "<div class='divCal' id='calendar'></div>";
}
function show_calendar_list($atts)
{
    $callW = $atts['calendarwidth'];
    $callH = $atts['calendarheight'];
    $view = $atts['view'];
    $url = plugin_dir_url(__DIR__) . "includes/json/allEvent.json";
```

```
$data = file_get_contents($url);
switch ($view) {
   case "D":
      $views = "{ listDay: { buttonText: 'day' }, }";
      $menu = "'";
      $defaul = "listDay";
      break;
   case "W":
      $views = "{ listWeek: { buttonText: 'week' }, }";
      $menu = "'";
      $defaul = "listWeek";
      break;
   case "M":
      $views = "{ listMonth: { buttonText: 'month' }, }";
      $menu = "'";
      $defaul = "listMonth";
      break;
   case "Y":
      $views = "{ listYear: { buttonText: 'year' }, }";
      $menu = "'";
      $defaul = "listYear";
      break;
   case "DW":
      $views = "{ listDay: { buttonText: 'day' }, listWeek: { buttonText: 'week' }, }";
      $menu = "'listDay,listWeek'";
      $defaul = "listWeek";
      break;
   case "WM":
     $views = "{listWeek: { buttonText: 'week' },listMonth: { buttonText: 'month' }, }";
      $menu = "'listWeek,listMonth'";
      $defaul = "listWeek";
      break;
   case "MY":
```

```
    $views = "{ listMonth: { buttonText: 'month' },listYear: {buttonText: 'year'}, }";
    $menu = "'listMonth,listYear'";
    $defaul = "listWeek";
    break;

case "DM":
    $views = "{ listDay: { buttonText: 'day' },listMonth: { buttonText: 'month' }, }";
    $menu = "'listDay,listMonth'";
    $defaul = "listDay";
    break;
case "DY":
    $views = "{ listDay: { buttonText: 'day' },listYear: { buttonText: 'year' }, }";
    $menu = "'listDay,listYear'";
    $defaul = "listDay";
    break;
case "WY":
    $views = "{listWeek: { buttonText: 'week' },listYear: { buttonText: 'year' }, }";
    $menu = "'listWeek,listYear'";
    $defaul = "listWeek";
    break;
case "DWM":
    $views = "{ listDay: { buttonText: 'day' }, listWeek: { buttonText: 'week' }, listMonth: { buttonText: 'month' }, }";
    $menu = "'listDay,listWeek,listMonth'";
    $defaul = "listWeek";
    break;
case "WMY":
    $views = "{ listWeek: { buttonText: 'week' }, listMonth: { buttonText: 'month' },listYear: {buttonText: 'year'}, }";
    $menu = "'listWeek,listMonth,listYear'";
    $defaul = "listWeek";
    break;
case "DMY":
    $views = "{ listDay: { buttonText: 'day' }, listMonth: { buttonText: 'month' },listYear: {buttonText: 'year'}, }";
    $menu = "'listDay,listMonth,listYear'";
```

```
        $defaul = "listMonth";
        break;
    case "DWY":
        $views = "{ listDay: { buttonText: 'day' }, listWeek: { buttonText: 'week' }, listYear: { buttonText: 'year' }, }";
        $menu = "'listDay,listWeek,listYear'";
        $defaul = "listWeek";
        break;
    case "DWMY":
        $views = "{ listDay: { buttonText: 'day' },listWeek: { buttonText: 'week' }, listMonth: { buttonText: 'month' },listYear: {buttonText: 'year'}, }";
        $menu = "'listDay,listWeek,listMonth,listYear'";
        $defaul = "listWeek";
        break;
    default:
        $views = "{ listDay: { buttonText: 'day' }, listWeek: { buttonText: 'week' }, listMonth: { buttonText: 'month' }, }";
        $menu = "'listDay,listWeek,listMonth'";
        $defaul = "listWeek";
  }
?>
  <script>
    document.addEventListener('DOMContentLoaded', function() {
      var calendarEl = document.getElementById('calendar');
      var calendar = new FullCalendar.Calendar(calendarEl, {
        plugins: ['list'],
        timeZone: 'UTC',
        defaultView: '<?php echo $defaul; ?>',
        views: <?php echo $views; ?>,
        header: {
          left: 'title',
          center: '',
          right: <?php echo $menu ?>
        },
        events: <?php echo "[" . $data . "]"; ?>,
```

```
            eventTimeFormat: {
                hour: '2-digit',
                minute: '2-digit',
                meridiem: false
            }
        });
        calendar.render();
    });
  </script>
  <style>
    .divCal {
        width: <?php echo $callW; ?>;
        height: <?php echo $callH; ?>;
    }
  </style>
<?php
  return "<div class='divCal' id='calendar'></div>";
}
function show_calendar_choose($atts)
{
  $callW = $atts['calendarwidth'];
  $callH = $atts['calendarheight'];
  $callL = $atts['locale'];
  $url = plugin_dir_url(__DIR__) . "includes/json/categorized";
  $countCats = $atts['countcats'];
  $data = "";
  for ($i = 1; $i <= $countCats; $i++) {
    ${"category" . $i} = $atts['cat' . $i];
    ${"url" . $i} = $url . "/" . ${"category" . $i} . ".json";
    ${"data" . $i} = file_get_contents(${"url" . $i});
    $data .= "" . ${"data" . $i} . "";
  }
?>
```

```
<script>
  document.addEventListener('DOMContentLoaded', function() {
    var calendarEl = document.getElementById('calendar');
    var calendar = new FullCalendar.Calendar(calendarEl, {
      plugins: ['dayGrid', 'timeGrid', 'interaction'],
      timeZone: 'local',
      eventLimit: true,
      nowIndicator: true,
      displayEventTime: false,
      locale: '<?php echo $callL ?>',
      header: {
        left: 'dayGridMonth,timeGridWeek,timeGridDay custom1',
        center: '',
        right: 'title',
      },
      footer: {
        left: 'today',
        center: '',
        right: 'custom2 prevYear,prev,next,nextYear'
      },
      events: <?php echo "[" . $data . "]"; ?>,
      eventTimeFormat: { // like '14:30'
        hour: '2-digit',
        minute: '2-digit',
        meridiem: false
      }
    });
    calendar.render();
  });
</script>
<style>
  .divCal {
    width: <?php echo $callW; ?>;
```

```
        height: <?php echo $callH; ?>;
      }
   </style>
<?php
   return  "<div class = 'divCal' id='calendar'></div>";
}
function show_calendar_list_choose($atts)
{
   $callW = $atts['calendarwidth'];
   $callH = $atts['calendarheight'];
   $callL = $atts['locale'];
   $view = $atts['view'];
   $url = plugin_dir_url(__DIR__) . "includes/json/categorized";
   $countCats = $atts['countcats'];
   $data = "";
   for ($i = 1; $i <= $countCats; $i++) {
      ${"category" . $i} = $atts['cat' . $i];
      ${"url" . $i} = $url . "/" . ${"category" . $i} . ".json";
      ${"data" . $i} = file_get_contents(${"url" . $i});
      $data .= "" . ${"data" . $i} . "";
   }
   switch ($view) {
      case "D":
         $views = "{ listDay: { buttonText: 'day' }, }";
         $menu = "'";
         $defaul = "listDay";
         break;
      case "W":
         $views = "{ listWeek: { buttonText: 'week' }, }";
         $menu = "'";
         $defaul = "listWeek";
         break;
      case "M":
```

```
    $views = "{ listMonth: { buttonText: 'month' }, }";
    $menu = "'";
    $defaul = "listMonth";
    break;
case "Y":
    $views = "{ listYear: { buttonText: 'year' }, }";
    $menu = "'";
    $defaul = "listYear";
    break;
case "DW":
    $views = "{ listDay: { buttonText: 'day' }, listWeek: { buttonText: 'week' }, }";
    $menu = "'listDay,listWeek'";
    $defaul = "listWeek";
    break;
case "WM":
  $views = "{listWeek: { buttonText: 'week' },listMonth: { buttonText: 'month' }, }";
    $menu = "'listWeek,listMonth'";
    $defaul = "listWeek";
    break;
case "MY":
   $views = "{ listMonth: { buttonText: 'month' },listYear: {buttonText: 'year'}, }";
    $menu = "'listMonth,listYear'";
    $defaul = "listWeek";
    break;
case "DM":
   $views = "{ listDay: { buttonText: 'day' },listMonth: { buttonText: 'month' }, }";
    $menu = "'listDay,listMonth'";
    $defaul = "listDay";
    break;
case "DY":
    $views = "{ listDay: { buttonText: 'day' },listYear: { buttonText: 'year' }, }";
    $menu = "'listDay,listYear'";
    $defaul = "listDay";
```

```
        break;
    case "WY":
        $views = "{listWeek: { buttonText: 'week' },listYear: { buttonText: 'year' }, }";
        $menu = "'listWeek,listYear'";
        $defaul = "listWeek";
        break;
    case "DWM":
        $views = "{ listDay: { buttonText: 'day' }, listWeek: { buttonText: 'week' }, listMonth: { buttonText: 'month' }, }";
        $menu = "'listDay,listWeek,listMonth'";
        $defaul = "listWeek";
        break;
    case "WMY":
        $views = "{ listWeek: { buttonText: 'week' }, listMonth: { buttonText: 'month' },listYear: {buttonText: 'year'}, }";
        $menu = "'listWeek,listMonth,listYear'";
        $defaul = "listWeek";
        break;
    case "DMY":
        $views = "{ listDay: { buttonText: 'day' }, listMonth: { buttonText: 'month' },listYear: {buttonText: 'year'}, }";
        $menu = "'listDay,listMonth,listYear'";
        $defaul = "listMonth";
        break;
    case "DWY":
        $views = "{ listDay: { buttonText: 'day' }, listWeek: { buttonText: 'week' }, listYear: { buttonText: 'year' }, }";
        $menu = "'listDay,listWeek,listYear'";
        $defaul = "listWeek";
        break;
    case "DWMY":
        $views = "{ listDay: { buttonText: 'day' },listWeek: { buttonText: 'week' }, listMonth: { buttonText: 'month' },listYear:
{buttonText: 'year'}, }";
        $menu = "'listDay,listWeek,listMonth,listYear'";
        $defaul = "listWeek";
        break;
    default:
```

```
        $views = "{ listDay: { buttonText: 'day' }, listWeek: { buttonText: 'week' }, listMonth: { buttonText: 'month' }, }";
        $menu = "'listDay,listWeek,listMonth'";
        $defaul = "listWeek";
    }
?>
    <script>
        document.addEventListener('DOMContentLoaded', function() {
            var calendarEl = document.getElementById('calendar');
            var calendar = new FullCalendar.Calendar(calendarEl, {
                plugins: ['list'],
                timeZone: 'UTC',
                defaultView: '<?php echo $defaul ?>',
                locale: '<?php echo $callL ?>',
                views: <?php echo $views ?>,
                header: {
                    left: 'title',
                    center: '',
                    right: <?php echo $menu ?>
                },
                events: <?php echo "[" . $data . "]"; ?>,
                eventTimeFormat: {
                    hour: '2-digit',
                    minute: '2-digit',
                    meridiem: false
                }
            });
            calendar.render();
        });
    </script>
    <style>
        .divCal {
            width: <?php echo $callW; ?>;
            height: <?php echo $callH; ?>;
```

```
    }
  </style>
<?php
  return "<div class='divCal' id='calendar'></div>";
}
```

config/classes/category.php

```
<?php
if (isset($_POST['addCat'])) {
    $category_obj = new Category();
    $catTitle =  sanitize_text_field($_POST['catTitle']);
    $catSlug = preg_replace('/\s+/', '-', $catTitle);
    $catDesc = sanitize_text_field($_POST['catDesc']);
    $catUrl = sanitize_text_field($_POST['catUrl']);
    $catColor = $_POST['catColor'];
    if (strpos($catTitle, '-') !== false) {
        array_push($errors, "Title can't contain \"-\"");
    }
    global $wpdb;
    $cat_table = $wpdb->prefix . 'wps_mc_categories';
    $checkSameTitle = $wpdb->get_var("SELECT COUNT(*) FROM $cat_table WHERE title = '$catTitle'");
    if ($checkSameTitle > 0) {
        array_push($errors, "Title already exist");
    }
    $checkSameColor = $wpdb->get_var("SELECT COUNT(*) FROM $cat_table WHERE color = '$catColor'");
    if ($checkSameColor > 0) {
        array_push($errors, "Color already assigned in another category");
    }
    if (count($errors) == 0) {
        $category_obj->addCategory($catTitle, $catSlug, $catDesc, $catUrl, $catColor);
        array_push($corrects, "Category: \"" . $catTitle . "\" added");
        $json_obj = new Json();
        $json_obj->updateAllCategoriesJSON();
        $json_obj->updateAllJSON();
    }
}
if (isset($_POST['updateCat'])) {
    $category_obj = new Category();
    $catTitle =  sanitize_text_field($_POST['categoryTitleUp']);
```

```
    $catSlug = preg_replace('/\s+/', '-', $catTitle);
    $catDesc = sanitize_text_field($_POST['categoryDescUp']);
    $catColor = $_POST['categoryColorUp'];
    $catUrl = $_POST['categoryUrlUp'];
    $catId = $_POST['updateCategoryId'];
    if (strpos($catTitle, '-') !== false) {
        array_push($errors, "Title can't contain \"-\"");
    }
    global $wpdb;
    $cat_table = $wpdb->prefix . 'wps_mc_categories';
    $checkSameTitle = $wpdb->get_var("SELECT COUNT(*) FROM $cat_table WHERE title = '$catTitle' AND id != '$catId'");
    if ($checkSameTitle > 0) {
        array_push($errors, "Title already exist");
    }
    $checkSameColor = $wpdb->get_var("SELECT COUNT(*) FROM $cat_table WHERE color = '$catColor' AND id != '$catId'");
    if ($checkSameColor > 0) {
        array_push($errors, "Color already assigned in another category");
    }
    if (count($errors) == 0) {
        $category_obj->updateCategory($catId, $catTitle, $catSlug, $catDesc, $catUrl, $catColor);
        array_push($corrects, "Category: \"" . $catTitle . "\" updated");
        $json_obj = new Json();
        $json_obj->updateAllCategoriesJSON();
        $json_obj->updateAllJSON();
    }
}
if (isset($_POST['deleteCat'])) {
    $catTitle = $_POST['deleteCatTitle'];
    $catID = $_POST['deleteCatID'];
    $eventobj = new Event();
    $eventobj->updateDeletedCatEvents($catID);
    $category_obj = new Category();
    $category_obj->deleteCategory($catID);
```

```
    array_push($corrects, "Category: \"" . $catTitle . " \" deleted");
    $json_obj = new Json();
    $json_obj->updateAllCategoriesJSON();
    $json_obj->updateAllJSON();
}
```

config/classes/events.php

```
<?php
if (isset($_POST['addEvent'])) {
    $event_obj = new Event();
    $eventTitle =  sanitize_text_field($_POST['eventTitle']);
    $eventDesc = sanitize_text_field($_POST['eventDesc']);
    $startDate = $_POST['startDate'];
    $endDate = $_POST['endDate'];
    $eventUrl = sanitize_text_field($_POST['eventUrl']);
    $startTime = $_POST['startTime'];
    $endTime = $_POST['endTime'];
    $eventLocation = sanitize_text_field($_POST['eventLocation']);
    $eventCat = $_POST['cat'];
    $startDate = date("Y-m-d", strtotime($startDate));
    $endDate = date("Y-m-d", strtotime($endDate));
    $startDate = str_replace('-', '/', $startDate);
    $endDate = str_replace('-', '/', $endDate);
    $start = $startDate . " " . $startTime;
    $end = $endDate . " " . $endTime;
    $startCount = preg_replace('/[^0-9]/', '', $start);
    $endCount = preg_replace('/[^0-9]/', '', $end);
    $duration = $endCount - $startCount;
    if ($duration < 0) {
        array_push($errors, "Start Date/Time must be before End Date/Time");
    }
    if (count($errors) == 0) {
        if ($eventCat == '0') {
            $event_obj->addEventWithoutCat($eventTitle, $eventDesc, $start, $end, $eventLocation, $eventUrl);
        } else {
            $event_obj->addEvent($eventTitle, $eventDesc, $start, $end, $eventLocation, $eventCat, $eventUrl);
        }
        array_push($corrects, "Event: \"" . $eventTitle . "\" added");
        $json_obj = new Json();
```

```
        $json_obj->updateAllCategoriesJSON();
        $json_obj->updateAllJSON();
    }
}
if (isset($_POST['deleteEvent'])) {
    $eventTitle = $_POST['deleteEventTitle'];
    $eventID = $_POST['deleteEventID'];
    $event_obj = new Event();
    $event_obj->deleteEvent($eventID);
    array_push($corrects, "Event: \"" . $eventTitle . "\" deleted");
    $json_obj = new Json();
    $json_obj->updateAllCategoriesJSON();
    $json_obj->updateAllJSON();
}
if (isset($_POST['updateEvent'])) {
    $event_obj = new Event();
    $eventTitle = sanitize_text_field($_POST['eventTitleUp']);
    $eventDesc = sanitize_text_field($_POST['eventDescUp']);
    $startDate = $_POST['startDateUp'];
    $endDate = $_POST['endDateUp'];
    $eventId = $_POST['updateEventId'];
    $startTime = $_POST['startTimeUp'];
    $endTime = $_POST['endTimeUp'];
    $eventLocation = sanitize_text_field($_POST['eventLocationUp']);
    $url = sanitize_text_field($_POST['eventUrlUp']);
    $eventCat = $_POST['catUp'];
    $startDate = date("Y/m/d", strtotime($startDate));
    $endDate = date("Y/m/d", strtotime($endDate));
    $start = $startDate . " " . $startTime;
    $end = $endDate . " " . $endTime;
    $startCount = preg_replace('/[^0-9]/', '', $start);
    $endCount = preg_replace('/[^0-9]/', '', $end);
    $duration = $endCount - $startCount;
```

```
    if ($duration < 0) {
array_push($errors, "The event wasn't updated: Start Date/Time must be before End Date/Time");
    }
    if (count($errors) == 0) {
        if ($eventCat == '0') {
            $eventCat = null;
            $event_obj->updateEvent($eventId, $eventTitle, $eventDesc, $start, $end, $eventLocation, $eventCat, $url);
            array_push($corrects, "Event: \"" . $eventTitle . "\" updated without cat");
        } else {
            $event_obj->updateEvent($eventId, $eventTitle, $eventDesc, $start, $end, $eventLocation, $eventCat, $url);
            array_push($corrects, "Event: \"" . $eventTitle . "\" updated with cat");
        }
        $json_obj = new Json();
        $json_obj->updateAllCategoriesJSON();
        $json_obj->updateAllJSON();
    }
}
```

config/queries/fetch_category.php

```
<?php
if (isset($_POST['get_the_cat'])) {

    $id = $_POST['get_the_cat'];
    $json_obj = new Json();
    $json_obj->createOneCatJSON($id);

}
```

config/queries/fetch_event.php

```
<?php
if (isset($_POST['get_the_event'])) {

    $id = $_POST['get_the_event'];
    $json_obj = new Json();
    $json_obj->createOneEventJSON($id);
}
```