



ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ
«SERVER DATA BACKUP MANAGEMENT»



Εικόνα 1.1: Τίτλος Διπλωματικής

Του φοιτητή
ΕΜΜΑΝΟΥΗΛΙΔΗ ΣΟΦΟΚΛΗ
Αρ. Μητρώου: 164659

Επιβλέπων
Ονοματεπώνυμο **ΟΥΓΙΑΡΟΓΛΟΥ**
ΣΤΕΦΑΝΟΣ
Βαθμίδα Καθηγητής

Ημερομηνία

Server Data Backup Management

Κωδικός Δ.Ε. ...

Εμμανουηλίδης Σοφοκλής

Ονοματεπώνυμο εισηγητή ...

Ημερομηνία ανάληψης Δ.Ε. ...

Ημερομηνία περάτωσης Δ.Ε. ...

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 παρούσα εργασία αποτελεί πνευματική ιδιοκτησία του φοιτητή Εμμανουηλίδη Σοφοκλή που την εκπόνησε. Στο πλαίσιο της πολιτικής ανοικτής πρόσβασης, ο συγγραφέας/δημιουργός εκχωρεί στο Διεθνές Πανεπιστήμιο της Ελλάδος άδεια χρήσης του δικαιώματος αναπαραγωγής, δανεισμού, παρουσίασης στο κοινό και ψηφιακής διάχυσης της εργασίας διεθνώς, σε ηλεκτρονική μορφή και σε οποιοδήποτε μέσο, για διδακτικούς και ερευνητικούς σκοπούς, άνευ ανταλλάγματος. Η ανοικτή πρόσβαση στο πλήρες κείμενο της εργασίας, δεν σημαίνει καθ' οιονδήποτε τρόπο παραχώρηση δικαιωμάτων διανοητικής ιδιοκτησίας του συγγραφέα/δημιουργού, ούτε επιτρέπει την αναπαραγωγή, αναδημοσίευση, αντιγραφή, πώληση, εμπορική χρήση, διανομή, έκδοση, μεταφόρτωση (downloading), ανάρτηση (uploading), μετάφραση, τροποποίηση με οποιονδήποτε τρόπο, τμηματικά ή περιληπτικά της εργασίας, χωρίς τη ρητή προηγούμενη έγγραφη συναίνεση του συγγραφέα/δημιουργού.

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

Αφιέρωση

Την πτυχιακή μου εργασία την αφιερώνω σε όλη μου την οικογένεια που όλα αυτά τα χρόνια είναι στο πλάι μου και με στηρίζει σε κάθε στιγμή της ζωής μου και εύχομαι η εργασία αυτή στο μέλλον να βοηθήσει όποιον την χρειαστεί στις εργασίες και απαιτήσεις τους και να τους διευκολύνει στην καθημερινότητα τους.

Πρόλογος

Ο κυριότερος λόγος που επέλεξα αυτό το θέμα της πτυχιακής είναι η αυξημένη ανάγκη στις μέρες μας για την ύπαρξη αντιγράφων ασφαλείας βάσεων δεδομένων. Στην σημερινή εποχή όπου όλα πλέον αρχίζουν να γίνονται ψηφιοποιημένα και να μειώνεται η χρήση του χάρτινου αρχείου, η απώλεια δεδομένων μπορεί να έχει σοβαρές επιπτώσεις και ατομικά στους χρήστες και ιδιαίτερα στις επιχειρήσεις. Η διαχείριση ενός αξιόπιστου συστήματος backup είναι καθημερινή και σημαντική υπόθεση για την αποφυγή αστοχιών συστήματος, κυβερνοεπιθέσεων ή ανθρώπινων λαθών.

Αυτόν τον στόχο επιτυγχάνει η δημιουργία μιας πλατφόρμας όπου οι χρήστες μπορούν να πραγματοποιούν πολλαπλά και ταυτόχρονα backup βάσεων δεδομένων από διαφορετικά domains και να αποθηκεύονται τόσο τα αρχεία που δημιουργούνται, όσο και τα στοιχεία αυθεντικοποίησης τους αποθηκευμένα με αποκρυπτογράφηση στην βάση δεδομένων budbserver του παρόχου nirea.iee.ihu.gr.

Αυτή η πλατφόρμα δίνει την δυνατότητα στους χρήστες να πραγματοποιούν πολλαπλά backups απο διαφορετικούς παρόχους και βάσεις δεδομένων με ένα κουμπί , γλιτώνοντας έτσι πολύτιμο χρόνο και πολλαπλά logins και ssh.

Περίληψη

Το θέμα - ζητούμενο της παρούσας πτυχιακής εργασίας είναι να μπορούν οι χρήστες μέσω μιας ιστοσελίδας να πραγματοποιούν πολλαπλά backups σε πολυάριθμες βάσεις δεδομένων από τους ίδιους ή και διαφορετικούς παρόχους, γλιτώνοντας έτσι την ξεχωριστή αυθεντικοποίηση για κάθε μια βάση δεδομένων και πάροχο αντίστοιχα και κερδίζοντας πολύτιμο χρόνο. Όλα αυτά τα στοιχεία σύνδεσης και αυθεντικοποίησης αποθηκεύονται σε ξεχωριστό για κάθε χρήστη προφίλ με μοναδικό email ως username για τον καθένα αντίστοιχα.

Τα κυριότερα εργαλεία που χρησιμοποιήθηκαν για την πραγματοποίηση των αντιγράφων ασφαλείας είναι:

- 1) MYSQLDUMP για την δημιουργία αντιγράφων ασφαλείας στις βάσεις δεδομένων mysql[36][39].
- 2) PG_DUMP για την δημιουργία αντιγράφων ασφαλείας στις βάσεις δεδομένων postgres..

Το δύσκολο κομμάτι στην υλοποίηση αυτής της πλατφόρμας ήταν οι διαφορές μεταξύ των απαιτήσεων του κάθε παρόχου στους τομείς της αυθεντικοποίησης και της δημιουργίας αρχείου backup. Προκειμένου να αντιμετωπίσω αυτή την ποικιλία διαφορετικών απαιτήσεων, δημιούργησα ένα σύστημα το οποίο υποστηρίζει ssh tunneling[48] για ασφαλή επικοινωνία, καθώς και την υποστήριξη διαφορετικών ειδών βάσεων δεδομένων (mysql , postgres).

Προκειμένου να πετύχω την αλληλεπίδραση της εφαρμογής με την βάση δεδομένων χρησιμοποίησα PHP[36][41] (PHP: Hypertext Preprocessor). Για την δομή και την οργάνωση του περιεχομένου της ιστοσελίδας αντίστοιχα χρησιμοποίησα HTML[44] (Hyper Text Markup Language) και για την εμφάνιση και τον σχεδιασμό του περιβάλλοντος χρήστη χρησιμοποίησα CSS[52] (Cascading Style Sheets).

Με την χρήση και τον συνδυασμό όλων των παραπάνω τεχνολογιών, δημιούργησα ένα λειτουργικό και δυναμικό σύστημα δημιουργίας αντιγράφων ασφαλείας βάσεων δεδομένων.

Abstract

The purpose of this THESIS project is to allow users to perform multiple backups of numerous databases from the same OR different providers through a website, both for saving the need for separate authentication for each database and provider respectively and saving valuable time. All of these login and authentication details are stored in a separate unique profile for each user with a unique email as username for each one of these profiles respectively.

The main tools that are used to perform the backups are:

- MYSQLDUMP for backing up the 'mysql' databases
- PG_DUMP for backing up the 'postgres' databases

The most challenging parts in creating this platform and making it operational were the numerous differences between the requirements of each provider in the areas of authentication and backup file creation. In order to address this variety of different requirements, I created a system that supports ssh tunneling[48] for secure communication and also supports different kinds of databases (mysql, postgres).

In order to achieve the interaction between the application and the database I used PHP[36][41] (PHP: Hypertext Preprocessor). For the structure and the organization of the website content respectively I used HTML[44] (Hyper Text Markup Language). For the appearance and the design of the UI I used CSS[52] (Cascading Style Sheets).

By using and combining all of the above technologies, I created a functional, dynamic and user friendly database backup system.

«BACKUP DATABASE SERVER (BUDS)»

«SOFOKLIS EMMANOUILIDIS»

Ευχαριστίες

Ένα τεράστιο ευχαριστώ οφείλω στον κ. Στέφανο Ουγιάρογλου, ο οποίος μέσα στην καρδιά του καλοκαιριού επικοινωνήσε μαζί μου, είδε το ενδιαφέρον μου για πτυχιακή εργασία, με ανέλαβε και σε όλη την διάρκεια της είναι παρών με συμβουλές και ιδέες για βελτιώσεις. Επίσης θα ήθελα να ευχαριστήσω και τον κ. Σιδηρόπουλο Αντώνιο για την επίλυση αποριών και εύρεση λύσεων σε κομμάτια του κώδικα του προγράμματος μεγάλης σημασίας για την λειτουργικότητα του.

Περιεχόμενα

Αφιέρωση	4
Πρόλογος	5
Περίληψη	6
Abstract	7
Ευχαριστίες	9
Περιεχόμενα	10
Κατάλογος Εικόνων	12
Κεφάλαιο 1ο: Ασφάλεια Δεδομένων	1
1.1 Εισαγωγή	1
1.2 Αντίγραφο Ασφαλείας Βάσεων Δεδομένων	1
1.3 Κίνητρο	3
1.4 Συνεισφορά	3
1.5 Οργάνωση εργασίας	4
Κεφάλαιο 2ο: Τύποι Αντιγράφων Ασφαλείας	5
2.1 Εισαγωγή	5
2.2 FULL BACKUP - Πλήρες Αντίγραφο Ασφαλείας	5
2.2.1 Δημιουργία full backup με την χρήση MySQL	5
2.2.2 Δημιουργία full backup με την χρήση PostgreSQL	6
2.3 INCREMENTAL BACKUP - Επαυξητικό Αντίγραφο Ασφαλείας	7
2.3.1 Δημιουργία incremental backup με την χρήση MySQL	8
2.3.2 Δημιουργία incremental backup με την χρήση PostgreSQL	9
2.4 DIFFERENTIAL BACKUP - Διαφορικό Αντίγραφο Ασφαλείας	10
2.4.1 Δημιουργία differential backup με την χρήση MySQL	10
2.4.2 Δημιουργία differential backup με την χρήση PostgreSQL	11
2.5 SNAPSHOT BACKUP - Αντίγραφο Ασφαλείας Στιγμιοτύπου	12
2.5.1 Δημιουργία snapshot backup με την χρήση MySQL	13
2.5.2 Δημιουργία snapshot backup με την χρήση PostgreSQL	14
Κεφάλαιο 3ο: Τεχνολογίες	16
3.1 HTML	16
3.1.1 Θετικά χαρακτηριστικά της HTML	16
3.1.2 Αρνητικά χαρακτηριστικά της HTML	16
3.1.3 Συμπέρασμα και αιτιολόγηση επιλογής της HTML	17
3.2 CSS	18
3.2.1 Θετικά χαρακτηριστικά της CSS	18
3.2.2 Αρνητικά χαρακτηριστικά της CSS	18
3.3 PHP	19
3.3.1 Θετικά χαρακτηριστικά της PHP	19
3.3.2 Αρνητικά χαρακτηριστικά της PHP	20
3.4 SQL	20
3.4.1 Θετικά χαρακτηριστικά της SQL	21
3.4.2 Αρνητικά χαρακτηριστικά της SQL	21
Κεφάλαιο 4ο: Σχεδίαση και Ανάπτυξη του BUDBServer (Backup Database Server)	23

4.1 Λειτουργικές Απαιτήσεις	23
4.2 Αρχιτεκτονική Εφαρμογής	24
4.2.1 Flowcharts (Διαγράμματα Ροής)	24
4.2.2 ER Diagram (Διάγραμμα Σχέσης Οντοτήτων)	32
4.3 Υλοποίηση του FRONT END	33
4.3.1 Κώδικας CSS	33
4.3.2 Κώδικας Javascript	40
4.3.3 Τεχνολογία AJAX	44
4.4 Συμπεράσματα Front End	46
4.5 Υλοποίηση του BACK END	46
4.5.1 Δημιουργία αντιγράφου ασφαλείας	50
4.6 Github Repository	55
Κεφάλαιο 5ο: Παρουσίαση BUDBServer	56
5.1 Εισαγωγή - Περιγραφή	56
5.2 Λειτουργίες της εφαρμογής	56
Κεφάλαιο 6ο: Συμπεράσματα και μελλοντικές επεκτάσεις	70
6.1 Συμπεράσματα	70
6.2 Μελλοντικές Επεκτάσεις	71
BIBΛΙΟΓΡΑΦΙΑ ΠΤΥΧΙΑΚΗΣ	72

Κατάλογος Εικόνων

Εικόνα 1.1: Τίτλος Διπλωματικής	1
Εικόνα 2.2.1.1: Σύνταξη εντολής mysqldump	5
Εικόνα 2.2.1.2: Εντολή mysqldump για όλες τις βάσεις	6
Εικόνα 2.2.1.3: Εντολή mysqldump για συγκεκριμένες βάσεις	6
Εικόνα 2.2.2.1: Σύνταξη εντολής pg_dump	6
Εικόνα 2.2.2.2: Σύνταξη εντολής psql	7
Εικόνα 2.3.1.1: Σύνταξη εντολής mysqlbinlog	8
Εικόνα 2.3.1.2: Σύνταξη εντολής επαναφοράς incremental backup	8
Εικόνα 2.3.2.1: Γραμμές ενεργοποίησης WAL Logs	9
Εικόνα 2.3.2.2: Σύνταξη εντολής pg_restore	9
Εικόνα 2.4.1.1: Σύνταξη εντολής mysqldump	11
Εικόνα 2.4.1.2: Σύνταξη εντολής mysqlbinlog	11
Εικόνα 2.4.2.1: Σύνταξη εντολής pg_basebackup	12
Εικόνα 2.4.2.2: Σύνταξη εντολής rsync	12
Εικόνα 2.5.1.1: Εντολή FLUSH TABLES	13
Εικόνα 2.5.1.2: Σύνταξη εντολής lvcreate	14
Εικόνα 2.5.2.1: Σύνταξη εντολής pg_start_backup	14
Εικόνα 2.5.2.2: Εντολή lvcreate	14
Εικόνα 2.5.2.3: Εντολή pg_stop_backup	14
Εικόνα 4.2.1.1: Διάγραμμα ροής LOGIN φόρμας	24
Εικόνα 4.2.1.2: Διάγραμμα ροής SIGN_UP φόρμας	25
Εικόνα 4.2.1.3: Διάγραμμα ροής FORGOT_PASSWORD φόρμας	26
Εικόνα 4.2.1.4: Διάγραμμα ροής NEW DATABASE CONNECTION φόρμας	27
Εικόνα 4.2.1.5: Διάγραμμα ροής MY CONNECTIONS φόρμας	28
Εικόνα 4.2.1.6: Διάγραμμα ροής MY BACKUPS φόρμας	29
Εικόνα 4.2.1.7: Διάγραμμα ροής DELETE USER φόρμας	30
Εικόνα 4.2.1.8: Διάγραμμα ροής εφαρμογής	31
Εικόνα 4.2.2.1: Διάγραμμα σχέσης οντοτήτων εφαρμογής (ER)	32
Σύνταξη layout με την χρήση του universal selector	33
Στήσιμο εμφάνισης σώματος HTML	34
Εικόνα 4.3.1: Εφέ Θαμπώματος	35
Σύνταξη κώδικα ιδιοτήτων εμφάνισης κλάσης .wrapper	35
Ιδιότητες εμφάνισης κουμπιών στην κλάση .btn	36
Ιδιότητες εμφάνισης disabled button	36
Εικόνα 4.3.2: Εμφάνιση enabled και disabled buttons	37
Ιδιότητες εμφάνισης των select boxes	39
Εικόνα 4.3.3: Select Box φόρμας Sign Up	39
CSS Media Queries	40
Χρήση Javascript για δυναμική επιβεβαίωση διαγραφής	41
Εικόνα 4.3.2.1: Φόρμα υποβολής κωδικού για διαγραφή σύνδεσης	42
Σύνταξη εντολών Javascript για εμφάνιση modal και διαγραφή σύνδεσης	43
Σύνταξη εντολών Javascript για απόκρυψη modal	43
Εικόνα 4.3.1: Χρήση AJAX με JAVASCRIPT	44

Χρήση AJAX για ασύγχρονη υποβολή κωδικού και confirm	45
Εικόνα 4.4.1: Διαχείριση αιτήματος POST	47
Εικόνα 4.4.2: Postman POST request	48
Εικόνα 4.4.3: Postman GET request	48
Εικόνα 4.4.4: Κώδικας υλοποίησης απάντησης αιτήματος GET	49
Εικόνα 4.4.5: Συνάρτηση αποκρυπτογράφησης κωδικών	50
SQL Statement για αναζήτηση στοιχείων σύνδεσης	50
Εικόνα 4.4.6: Στοιχεία σύνδεσης με αποκρυπτογράφηση κωδικών πρόσβασης	51
Εικόνα 4.4.7: Δημιουργία φακέλου σύνδεσης	51
Εικόνα 4.4.8: Δημιουργία ονόματος αντιγράφου ασφαλείας	52
Σύνταξη εντολής δημιουργίας backup MySQL βάσης	52
Σύνταξη εντολής δημιουργίας backup MySQL βάσης	53
Σύνταξη εντολής εκτέλεσης exec();	53
Σύνταξη εντολής εκτέλεσης exec();	54
Εικόνα 5.2.1: Φόρμα Login	56
Εικόνα 5.2.2: Φόρμα Sign Up	57
Εικόνα 5.2.3: Φόρμα επαναφοράς κωδικού πρόσβασης, προσθήκη email	58
Εικόνα 5.2.4: Φόρμα υποβολής απάντησης ασφαλείας και δημιουργίας κωδικού πρόσβασης	59
Εικόνα 5.2.5: Αρχική σελίδα χρήστη	60
Εικόνα 5.2.6: Φόρμα δημιουργίας καινούργιας σύνδεσης	61
Εικόνα 5.2.7: Υποβολή στοιχείων για σύνδεση SSH	62
Εικόνα 5.2.8: Σενάριο λάθους ονόματος βάσης	63
Εικόνα 5.2.9: Σενάριο λάθους κωδικού σύνδεσης SSH	63
Εικόνα 5.2.10: Σενάριο υποβολής σωστών στοιχείων αυθεντικοποίησης	64
Εικόνα 5.2.11: Φόρμα αποθηκευμένων συνδέσεων χρήστη	65
Εικόνα 5.2.12: Dropdown Box επιλογών είδους αντιγράφου ασφαλείας	65
Εικόνα 5.2.13: Μήνυμα επιτυχούς δημιουργίας αντιγράφου ασφαλείας	66
Εικόνα 5.2.14: Υποβολή κωδικού για διαγραφή σύνδεσης	67
Εικόνα 5.2.15: Φόρμα αποθηκευμένων αντιγράφων ασφαλείας χρήστη	68
Εικόνα 5.2.16: Διαδικασία διαγραφής λογαριασμού (υποβολή κωδικού πρόσβασης)	69
Εικόνα 5.2.17: Διαδικασία διαγραφής λογαριασμού (επιβεβαίωση διαγραφής)	70

Κεφάλαιο 1ο: Ασφάλεια Δεδομένων

1.1 Εισαγωγή

[1] [3] Στον σύγχρονο ψηφιακό κόσμο, σταδιακά αρχίζει να μειώνεται η χρήση του χαρτιού και γενικότερα του υλικού μέσου αποθήκευσης οποιουδήποτε είδους πληροφορίας, δεδομένου, αρχείου κτλ. Τα δεδομένα πλέον αποθηκεύονται ψηφιακά σε βάσεις δεδομένων, παρέχοντας έτσι την δυνατότητα στον κόσμο να έχει ευκολότερη και γρηγορότερη πρόσβαση στις πληροφορίες. Ταυτόχρονα αυξάνεται σε μεγάλο βαθμό η εξοικονόμηση του χώρου.

[2] Παρότι όμως υπάρχουν αρκετά οφέλη με την ψηφιακή αποθήκευση δεδομένων, η ανάγκη για την ύπαρξη αντιγράφων ασφαλείας παραμένει σημαντική και αμετάβλητη. Όπως παλαιότερα διατηρούσαμε φυσικά αντίγραφα για προστασία από οποιαδήποτε μορφή απώλειας, έτσι και σήμερα η δημιουργία αντιγράφων ασφαλείας (backup) είναι απαραίτητη για την ασφάλεια των ψηφιακών δεδομένων.

Η απώλεια ή ακόμα και η καταστροφή ψηφιακών αρχείων, η δυσλειτουργία και η μη προσβασιμότητα σε βάσεις δεδομένων μπορεί να έχει σοβαρές συνέπειες είτε για τον χρήστη ατομικά είτε και για ολόκληρες επιχειρήσεις, καθιστώντας έτσι την ύπαρξη αντιγράφων ασφαλείας κρίσιμα για την επιχειρησιακή συνέχεια και την προστασία των πληροφοριών και δεδομένων.

1.2 Αντίγραφα Ασφαλείας Βάσεων Δεδομένων

Τα 2 βασικά εργαλεία που χρησιμοποιήθηκαν για την δημιουργία αντιγράφων ασφαλείας στις βάσεις δεδομένων **MySQL**[36][39] και **PostgreSQL**[38], είναι το **mysqldump** και το **pg_dump** αντίστοιχα. Καθένα από αυτά τα εργαλεία έχει τα δικά του πλεονεκτήματα και μειονεκτήματα, ο συνδυασμός των οποίων επηρεάζουν την επιλογή τους από τον κάθε χρήστη, καθοριζόμενη από τις εκάστοτε ανάγκες του.

- **mysqldump**: Είναι σημαντικό μέρος του συνόλου εντολών της MySQL[36][39] και χρησιμοποιείται για την δημιουργία λογικών αντιγράφων ασφαλείας. Μέσω του mysqldump, οι χρήστες μπορούν να εξάγουν δομές και δεδομένα βάσεων δεδομένων σε μορφή SQL, έχοντας έτσι την δυνατότητα αναπαραγωγής τους σε έναν άλλο διακομιστή ή απλά για σκοπούς αποθήκευσης. [4]
- **pg_dump**: Είναι το αντίστοιχο εργαλείο για την PostgreSQL, επιτρέποντας την δημιουργία αντιγράφων ασφαλείας μιας βάσης δεδομένων σε μορφή SQL script. Το pg_dump εξασφαλίζει ότι τα αντίγραφα θα είναι πλήρη και ενημερωμένα, ακόμα και όταν η βάση δεδομένων χρησιμοποιείται ενεργά. [5]

ΠΛΕΟΝΕΚΤΗΜΑΤΑ

- **mysqldump:**
 - Ευκολία χρήσης και ευρεία υποστήριξης.
 - Δυνατότητα εξαγωγής δεδομένων αντιγράφου ασφαλείας σε διάφορες μορφές, όπως SQL, CSV και XML.
 - Κατάλληλο για χρήση σε μικρές έως μεσαίες βάσεις δεδομένων.[7]
- **pg_dump:**
 - Δημιουργία συνεπών αντιγράφων, ακόμη και κατά την διάρκεια ενεργής χρήσης της βάσης δεδομένων.
 - Υποστήριξη διαφόρων μορφών εξαγωγής, όπως plain-text SQL script και archive files.[5][6]
 - Επιτρέπει την επιλεκτική ανάκτηση δεδομένων μέσω της εντολής pg_restore.[6]

ΜΕΙΟΝΕΚΤΗΜΑΤΑ

- **mysqldump:**
 - Για πολύ μεγάλες βάσεις δεδομένων μπορεί να είναι αργό λόγω της μονοπυρηνικής επεξεργασίας δεδομένων.
 - Αντίστοιχα η διαδικασία ανάκτησης δεδομένων μπορεί να είναι ιδιαίτερα χρονοβόρα για πολύ μεγάλα σύνολα δεδομένων.[7]
- **pg_dump:**
 - Μπορεί να δημιουργήσει αντίγραφο ασφαλείας για μόνο μια βάση δεδομένων και όχι για πολλές ταυτόχρονα.[9]
 - Η δημιουργία αντιγράφων ασφαλείας ενός συστήματος αρχείων λειτουργεί μόνο για την πλήρη δημιουργία αντιγράφων ασφαλείας και επαναφορά ολόκληρης συστάδας βάσεων δεδομένων. Αυτό σημαίνει ότι η δημιουργία αντιγράφων ασφαλείας και η επαναφορά ορισμένων μεμονωμένων αρχείων ή πινάκων αντίστοιχα είναι αδύνατη.[8]

Λαμβάνοντας υπόψη όλα τα παραπάνω δεδομένα, καταλήγουμε στο συμπέρασμα ότι η επιλογή μεταξύ mysqldump και pg_dump εξαρτάται από τις ανάγκες του κάθε συστήματος και τα χαρακτηριστικά της κάθε βάσης δεδομένων. Η Mysql είναι προτιμότερο να χρησιμοποιείται για μικρότερες βάσεις δεδομένων και είναι πιο απλή και γρήγορη. Ωστόσο όταν πρόκειται για μεγαλύτερες βάσεις δεδομένων και για πιο πολύπλοκα δεδομένα, εκεί προτιμάται η χρήση της PostgreSQL ,η οποία προσφέρει συνεπή backups ακόμα και σε ενεργά συστήματα. Η διαχείριση της ωστόσο απαιτεί μεγαλύτερη κατανόηση και είναι πιο δύσκολη στη χρήση. Συνεπώς, η επιλογή εξαρτάται από το μέγεθος, την πολυπλοκότητα αλλά και τις ανάγκες ανάκτησης των δεδομένων.

1.3 Κίνητρο

Το κίνητρο μου για να δημιουργήσω αυτή την web εφαρμογή ήταν να διευκολύνω την δημιουργία αντιγράφων ασφαλείας από πολυάριθμες βάσεις δεδομένων από διαφορετικούς παρόχους , χωρίς να απαιτείται από τον χρήστη να συνδέεται ξεχωριστά με πολύπλοκες διαδικασίες σε κάθε μια από αυτές. Το μέγεθος του κινδύνου που μπορεί να επιφέρει στους χρήστες και στις επιχειρήσεις αντίστοιχα η απώλεια δεδομένων σε συνδυασμό με την ανάγκη για την ύπαρξη ενός πιο γρήγορου τρόπου δημιουργίας αντιγράφων, ήταν 2 σύγχρονα προβλήματα που ήθελα να τα λύσω με αυτή την εφαρμογή.

1.4 Συνεισφορά

Η τελική συνεισφορά της δημιουργίας μια web εφαρμογής, η οποία δημιουργεί backups βάσεων δεδομένων σε ξεχωριστά προφίλ είναι η εξασφάλιση της ταχύτητας, της ευκολίας και ειδικότερα της ασφάλειας των δεδομένων. Αυτή η εφαρμογή δίνει την δυνατότητα σε κάθε χρήστη να έχει απομακρυσμένη πρόσβαση στα δικά του αντίγραφα ασφαλείας,, χωρίς να χρειαστεί να ανησυχεί για απώλειες ή για περίπλοκες διαδικασίες ανάκτησης. Αυτό συμβάλλει και στην μείωση του χρόνου αποκατάστασης των δεδομένων, καθώς και στην καλύτερη οργάνωση και εμπειρία του χρήστη. Επιπλέον η αυτοματοποίηση της διαδικασίας της αυθεντικοποίησης για άμεση πρόσβαση στις βάσεις δεδομένων του χρήστη μειώνει σημαντικά τον χρόνο που απαιτείται για όλη την διαδικασία αυτή και ταυτόχρονα εξασφαλίζει ότι τα δεδομένα θα είναι πάντα διαθέσιμα, βελτιώνοντας έτσι την αξιοπιστία του συστήματος και μειώνοντας το ανθρώπινο λάθος σε μεγάλο βαθμό.

1.5 Οργάνωση εργασίας

Η εργασία αυτή οργανώνεται από τα στάδια που θα αναλυθούν παρακάτω, όπου με μια αλληλουχία βημάτων αναπτύχθηκε μια εφαρμογή δημιουργίας αντιγράφων ασφαλείας βάσεων δεδομένων.

Αρχικά, υλοποιήθηκαν οι βασικές λειτουργίες αυθεντικοποίησης χρηστών, με τον συνδυασμών των λειτουργιών σύνδεσης υπάρχοντος χρήστη (Login), εγγραφής καινούργιου χρήστη (Sign Up) και επαναφοράς κωδικού πρόσβασης (Forgot Password Reset) σε μία φόρμα. Έτσι εξασφαλίστηκε η ασφαλής και η άμεση πρόσβαση στην εφαρμογή και στα δεδομένα του κάθε χρήστη ξεχωριστά.

Στην συνέχεια αναπτύχθηκε η φόρμα δημιουργίας νέων συνδέσεων (New Database Connection), όπου ο χρήστης μπορεί να αποθηκεύει στοιχεία αυθεντικοποίησης για διαφορετικές τοπικές και απομακρυσμένες (μέσω ssh tunneling[48]) βάσεις δεδομένων. Ταυτόχρονα ο χρήστης μπορεί κατά την δημιουργία της καινούργιας σύνδεσης και το είδος της βάσης που επιθυμεί να αποθηκεύσει (PostgreSQL[38] , MySQL[36][39]) , για να χρησιμοποιηθούν οι αντίστοιχες εντολές δημιουργίας αντιγράφων ασφαλείας pg_dump και mysqldump.

Προτού ωστόσο αποθηκευτούν όλες αυτές οι διαφορετικές συνδέσεις, πραγματοποιούνται δοκιμές εγκυρότητας για να διασφαλιστεί η σωστή λειτουργία των συνδέσεων και να αποφευχθούν τα λάθη στις απόπειρες δημιουργίας αντιγράφων ασφαλείας.

Η αποθήκευση των αντιγράφων ασφαλείας που δημιουργεί ο χρήστης με το που καταφέρει να αποθηκεύσει επιτυχώς τις συνδέσεις του γίνεται στην βάση δεδομένων της ιστοσελίδας στο αντίστοιχο του προφίλ με την δυνατότητα και για τοπική αποθήκευση, διασφαλίζοντας τον χρήστη ότι μπορεί να τα κατεβάσει όποτε επιθυμεί.

Τέλος, όλο αυτό το περιβάλλον περιήγησης χρήστη εμπλουτίστηκε με την χρήση CSS[52], κάνοντας το πιο ελκυστικό και ευχάριστο, με ευδιάκριτα κουμπιά και ομαλή διάταξη των δεδομένων στην οθόνη του χρήστη. Παράλληλα, συνδυάζοντας HTML[44] και PHP[36][41] και με την χρήση των εντολών SQL για την επικοινωνία με τους servers, δημιουργήθηκε μια δομημένη και λειτουργική εφαρμογή, προσφέροντας ομαλή εμπειρία χρήστη και εξασφαλίζοντας του την δυνατότητα να κάνει εύκολα και γρήγορα backups και να τα διαχειρίζεται και να τα αποθηκεύει όπως και όπου αυτός επιθυμεί.

Κεφάλαιο 2ο: Τύποι Αντιγράφων Ασφαλείας

2.1 Εισαγωγή

Ανάλογα με τις ανάγκες του κάθε συστήματος και των χρηστών την κάθε χρονική στιγμή, υπάρχουν 4 διαφορετικά είδη αντιγράφων ασφαλείας. Αυτά τα 4 είδη backup (Full, Incremental, Differential και Snapshot) επιφέρουν διαφορετικά πλεονεκτήματα και μειονεκτήματα αντίστοιχα. Αυτές εφαρμόζονται και στην MySQL[36][39] και στην PostgreSQL[38] αντίστοιχα αλλά με διαφορετικούς μηχανισμούς. Στην MySQL χρησιμοποιούνται τα binary logs , ενώ στην PostgreSQL χρησιμοποιούνται τα WAL αρχεία. Παρακάτω θα αναλυθούν λεπτομερώς τα 4 είδη των backup και για τους 2 τύπους βάσεων δεδομένων.[10][11]

2.2 FULL BACKUP - Πλήρες Αντίγραφο Ασφαλείας

Το **Full Backup** είναι η βασικότερη και πιό ολοκληρωμένη μέθοδος δημιουργίας αντιγράφων ασφαλείας ανάμεσα σε όλα τα είδη των backups. Αποθηκεύει ολόκληρη τη βάση δεδομένων, περιλαμβάνοντας όλα τα δεδομένα αλλά και τη δομή της. Αυτή η μέθοδος είναι η πιο αξιόπιστη, καθώς παρέχει ένα πλήρες αντίγραφο της βάσης, το οποίο μπορεί να αποκατασταθεί εύκολα χωρίς να απαιτεί την χρήση ή τον συνδυασμό προηγούμενων αντιγράφων ασφαλείας. Παρόλα αυτά, καταναλώνει περισσότερο χρόνο και χώρο αποθήκευσης σε σχέση με άλλες μεθόδους, όπως το **incremental**. Συνήθως χρησιμοποιείται στις περιπτώσεις που απαιτείται πλήρης αποκατάσταση και σταθερότητα των δεδομένων των βάσεων. Αυτή η μέθοδος υποστηρίζεται και στην MySQL και στην PostgreSQL.

2.2.1 Δημιουργία full backup με την χρήση MySQL

Στην **MySQL** [36][39][12] , το πλήρες αντίγραφο βάσης δεδομένων μπορεί να δημιουργηθεί με την εντολή `mysqldump`.

```
$> mysqldump [arguments] > file_name
```

Εικόνα 2.2.1.1: Σύνταξη εντολής mysqldump

Αν ο χρήστης επιθυμεί να δημιουργήσει πλήρη αντίγραφα όλων των βάσεων δεδομένων, τότε συντάσει την εντολή με αυτόν τον τρόπο:

```
$> mysqldump --all-databases > dump.sql
```

Εικόνα 2.2.1.2: Εντολή mysqldump για όλες τις βάσεις

Στην περίπτωση που ο χρήστης επιθυμεί να δημιουργήσει αντίγραφα για συγκεκριμένες βάσεις δεδομένων, τότε συντάζει την εντολή με την χρήση της επιλογής --databases στην γραμμή εντολών.

```
$> mysqldump --databases db1 db2 db3 > dump.sql
```

Εικόνα 2.2.1.3: Εντολή mysqldump για συγκεκριμένες βάσεις

[14] Με την χρήση της mysqldump εντολής η διαδικασία δημιουργίας ενός πλήρες αντιγράφου ασφαλείας είναι ευκολότερη και απλούστερη, λόγω της δυνατότητα αποθήκευσης σε ένα απλό αρχείο sql, προσφέροντας έτσι την δυνατότητα επαναφοράς ολόκληρης της βάσης σε ένα συγκεκριμένο χρονικό σημείο σε περίπτωση απώλειας ή καταστροφής. Έτσι επιτυγχάνεται η καλύτερη δυνατή προστασία και πλήρη αποκατάσταση των δεδομένων, επειδή η καταγραφή αυτή περιλαμβάνει τα πάντα, συμπεριλαμβανομένων όλων των αρχείων δεδομένων, των αρχείων διαμόρφωσης καθώς και των δυαδικών αρχείων καταγραφής.

Ωστόσο υπάρχουν και μειονεκτήματα, με το μεγαλύτερο από αυτά να είναι ο μεγάλος χρόνος της δημιουργίας του πλήρες αντιγράφου, ειδικότερα για μεγάλες βάσεις δεδομένων με μεγάλο αριθμό δεδομένων. Ταυτόχρονα επηρεάζεται και ο αποθηκευτικός χώρος σημαντικά και δεν είναι ιδανικό για συστήματα που απαιτούν συχνές αποθηκεύσεις, με την επιβάρυνση των πόρων του συστήματος να αποτελεί άλλο ένα σημαντικό μειονέκτημα.

2.2.2 Δημιουργία full backup με την χρήση PostgreSQL

Στην PostgreSQL αντίστοιχα [13], χρησιμοποιείται η εντολή pg_dump και συντάσσεται ως εξής:

```
pg_dump dbname > dumpfile
```

Εικόνα 2.2.2.1: Σύνταξη εντολής pg_dump

Έπειτα για την επαναφορά του αρχείου που δημιουργείται με την pg_dump, χρησιμοποιείται η εντολή psql:



Εικόνα 2.2.2.2: Σύνταξη εντολής psql

, όπου στην παραπάνω γραμμή εντολών το dumpfile είναι το αρχείο που δημιουργήθηκε από την pg_dump.

Πριν την αποκατάσταση ενός πλήρες αντιγράφου ασφαλείας, ο χρήστης θα πρέπει να έχει όλα τα δικαιώματα για όλα τα δεδομένα και τα αντικείμενα που περιλαμβάνονται στο αντίγραφο ασφαλείας. Εάν αυτά δεν υπάρχουν, η πλήρη επαναφορά της βάσης θα αποτύχει.

Μπορεί να αποθηκεύσει τόσο την δομή της βάσης όσο και τα δεδομένα της σε διάφορες μορφές αρχείων, όπως αρχεία sql, αρχεία tar η και σε custom format. Ένα από τα μεγαλύτερα πλεονεκτήματα του είναι ότι μπορεί να εκτελεστεί από οποιαδήποτε απομακρυσμένο σύστημα έχει πρόσβαση στην εκάστοτε βάση δεδομένων, χωρίς να διακόπτει τις λειτουργίες της. Επιπροσθέτως, τα αρχεία που δημιουργούνται είναι συμβατά με νεότερες εκδόσεις PostgreSQL, διευκολύνοντας έτσι τόσο την αναβάθμιση των δεδομένων όσο και την μεταφορά τους μεταξύ διαφορετικών αρχιτεκτονικών συστημάτων.

Ωστόσο η χρήση της pg_dump επιφέρει και αυτή μειονεκτήματα, από τον ιδιαίτερα αυξημένο χρόνο που απαιτείται για την αποκατάσταση μεγάλων βάσεων δεδομένων μέχρι και του συνόλου των ειδικών δικαιωμάτων που απαιτούνται για την πλήρη εξαγωγή όλων των δεδομένων. Αν ο χρήστης δεν έχει τις απαραίτητες άδειες (SUPERUSER), περιορίζεται στην δημιουργία backups μόνο συγκεκριμένων κομματιών της εκάστοτε βάσης, με την χρήση των επιλογών στην γραμμή εντολών όπως η “-n schema” ή “-t table”. Επιπλέον κάποιες λειτουργίες που απαιτούν αποκλειστική πρόσβαση του χρήστη στην βάση δεδομένων, όπως για παράδειγμα συγκεκριμένες ALTER TABLE εντολές, μπορεί να επηρεάσουν σημαντικά την όλη διαδικασία.

2.3 INCREMENTAL BACKUP - Επauξητικό Αντίγραφο Ασφαλείας

Το **Incremental Backup** είναι μια μέθοδος δημιουργίας αντιγράφων ασφαλείας που αποθηκεύει μόνο τις αλλαγές που έχουν γίνει μετά το τελευταίο backup, περιορίζοντας σημαντικά την αποθήκευση μεγάλου όγκου δεδομένων. Σε αντίθεση με το full backup που περιλαμβάνει ολόκληρη την βάση, το incremental backup καταγράφει μόνο τις νέες ή τροποποιημένες εγγραφές, επιταχύνοντας την διαδικασία και ταυτόχρονα εξοικονομώντας πόρους, κάτι που το καθιστά την ιδανικότερα επιλογή για μεγάλες βάσεις δεδομένων.

Ωστόσο η διαδικασία αποκατάστασης αυτόματα γίνεται πιο περίπλοκη, καθώς απαιτείται η χρήση και ο συνδυασμός πολλών αντιγράφων ασφαλείας για επιτυχή αποκατάσταση όλων των δεδομένων. Αυτή η μέθοδος υποστηρίζεται και στην MySQL[36][39] και στην PostgreSQL[38].

2.3.1 Δημιουργία incremental backup με την χρήση MySQL

Μέσω της χρήσης των binary logs επιτυγχάνεται η δημιουργία ενός incremental backup στη MySQL. Αυτό συμβαίνει επειδή τα logs καταγράφουν όλες τις αλλαγές που πραγματοποιούνται στην εκάστοτε βάση δεδομένων. Αυτή η μέθοδος δημιουργίας αντιγράφων ασφαλείας επιτρέπει την αποθήκευση μόνο των διαφοροποιήσεων από το τελευταίο οποιουδήποτε είδους backup που πραγματοποιήθηκε, μειώνοντας έτσι σημαντικά τον χρόνο και τον αποθηκευτικό χώρο που απαιτείται σε σύγκριση με ένα πλήρες backup.

Προκειμένου να μπορούν να πραγματοποιηθούν τα incremental backups θα πρέπει τα binary logs να είναι ενεργοποιημένα στον MySQL server, ρυθμίζοντας την παράμετρο *log_bin* στο αρχείο ρυθμίσεων.

Έπειτα εφόσον υπάρχει το πλήρες backup αρχείο, χρησιμοποιούμε την εντολή *mysqlbinlog* προκειμένου να εξαχθούν όλες οι αλλαγές που καταγράφηκαν στα binary logs μετά το τελευταίο backup ως εξής:

```
>mysqlbinlog --read-from-remote-server --host=my_host --user=my_username --password --start-datetime="YYYY-MM-DD HH:MM:SS" > incremental_backup.sql
```

Εικόνα 2.3.1.1: Σύνταξη εντολής *mysqlbinlog*

Μετά την επιτυχή εξαγωγή των αλλαγών, με την χρήση της *mysql* κάνουμε επαναφορά του incremental backup ως εξής:

```
>mysql --user=your_user --password < incremental_backup.sql
```

Εικόνα 2.3.1.2: Σύνταξη εντολής επαναφοράς incremental backup

Ένα μεγάλο πλεονέκτημα των επαυξητικών αντιγράφων ασφαλείας είναι η σημαντική μείωση του χρόνου καθώς και του χώρου αποθήκευσης που απαιτούνται, λόγω του ότι αποθηκεύονται μόνο οι αλλαγές από το τελευταίο backup. Ωστόσο για την σωστή επαναφορά του backup απαιτείται η εφαρμογή του πλήρους backup και όλων των επακόλουθων incremental backups με την σωστή σειρά. Ιδιαίτερη προσοχή πρέπει να δίνουμε και στην αποθήκευση και διαχείριση των binary logs προκειμένου να διασφαλιστεί η ακεραιότητα των δεδομένων κατά την αποκατάσταση.[16]

2.3.2 Δημιουργία incremental backup με την χρήση PostgreSQL

Στην PostgreSQL η διαδικασία δημιουργίας incremental backups είναι λίγο διαφορετική, καθώς εδώ χρησιμοποιούμε το Write-Ahead Logging (WAL), το οποίο καταγράφει όλες τις αλλαγές που γίνονται στην εκάστοτε βάση δεδομένων. Επειδή η εντολή `pg_dump`, που είναι η βασική εντολή στην PostgreSQL για την δημιουργία αντιγράφων ασφαλείας, δεν υποστηρίζει incremental backups, χρησιμοποιούμε εργαλεία όπως το `pg_basebackup`, `pg_restore` και το WAL archiving.

[17]Προκειμένου να δημιουργηθούν incremental backups με WAL, πρέπει πρώτα να ενεργοποιηθεί η καταγραφή των WAL logs με την προσθήκη στο αρχείο `postgresql.conf` των εξής γραμμών:

```
archive_command = 'pgbackrest --stanza=demo-alt archive-push %p'
archive_mode = on
max_wal_senders = 3
wal_level = replica
```

Εικόνα 2.3.2.1: Γραμμές ενεργοποίησης WAL Logs

Έτσι αποθηκεύονται όλα τα WAL αρχεία στον φάκελο `/var/lib/postgresql/wal_archive/`.

Τα WAL αρχεία που είχαν καταγραφεί μετά το πλήρες backup μπορούν να επαναφερθούν με την χρήση της `pg_restore` ως εξής:

```
>pg_restore -d mydb /var/lib/postgresql/wal_archive/
```

Εικόνα 2.3.2.2: Σύνταξη εντολής `pg_restore`

Με την χρήση αυτής της μεθόδου δημιουργίας επαυξητικών αντιγράφων ασφαλείας επιτυγχάνεται αυξημένη αποδοτικότητα και ελαχιστοποίηση του αποθηκευτικού χώρου. Παρόλα αυτά υπάρχει και αυξημένη πολυπλοκότητα διαχείρισης των WAL logs και πρέπει να υπάρχει προσεκτικός συγχρονισμός με το πλήρες backup. [18]

2.4 DIFFERENTIAL BACKUP - Διαφορικό Αντίγραφο Ασφαλείας

[19]Το differential backup είναι μια μέθοδος δημιουργία αντιγράφων ασφαλείας που αποθηκεύει όλες τις αλλαγές που έχουν γίνει από το **τελευταίο πλήρες αντίγραφο**, σε αντίθεση με το incremental, όπου εκεί κάθε αντίγραφο περιλαμβάνει μόνο τις αλλαγές από το τελευταίο backup (είτε πλήρες είτε incremental). Αυτή η μέθοδος μειώνει τον χρόνο επαναφοράς, καθώς απαιτεί μόνο το αρχικό πλήρες backup και το πιο πρόσφατο differential backup. Ωστόσο, όσο αυξάνεται ο αριθμός των differential backups που δημιουργούνται τόσο αυξάνεται και το μέγεθος του καθενός, καθώς περιλαμβάνουν όλες τις αλλαγές από το τελευταίο πλήρες backup.

2.4.1 Δημιουργία differential backup με την χρήση MySQL

Στην MySQL, η δημιουργία differential backup επιτυγχάνεται μέσω της χρήσης των binary logs, στα οποία καταγράφονται όλες οι τροποποιήσεις στην εκάστοτε βάση δεδομένων. Αυτά τα αρχεία περιέχουν όλες τις απαιτούμενες, για την εύκολη αποκατάσταση δεδομένων σε συγκεκριμένο χρονικό σημείο, αλλαγές που έγιναν μετά το τελευταίο πλήρες αντίγραφο ασφαλείας.

[19]Πρώτα πρέπει να έχει δημιουργηθεί ένα πλήρες αντίγραφο ασφαλείας και να ξεκινήσει ένα νέο binary log με τον παρακάτω τρόπο:

```
mysqldump -u username -p --all-databases --single-transaction --flush-logs --master-data=2 > full_backup.sql
```

Εικόνα 2.4.1.1: Σύνταξη εντολής mysqldump

, όπου στην παραπάνω εντολή το --flush-logs αναγκάζει τον server να δημιουργήσει ένα νέο binary log, ενώ η --master-data=2 καταγράφει τη θέση του δυαδικού αρχείου καταγραφής στο αρχείο αντιγράφων ασφαλείας.

Μετά την δημιουργία του πλήρες backup, τα binary logs καταγράφουν κανονικά τις αλλαγές. Προκειμένου να δημιουργηθεί ένα διαφορικό αντίγραφο ασφαλείας, απαιτείται η χρήση της εντολής mysqlbinlog για να εξαχθούν οι αλλαγές από το τελευταίο πλήρες backup με την σύνταξη της εντολής ως εξής:

```
mysqlbinlog --start-position=123456 mysql-bin.000002 > differential_backup.sql
```

Εικόνα 2.4.1.2: Σύνταξη εντολής mysqlbinlog

,όπου στην θέση του 123456 τοποθετούμε την θέση του τελευταίου backup και στην θέση του mysql-bin.000002 τοποθετούμε το όνομα του αντίστοιχου binary log.

2.4.2 Δημιουργία differential backup με την χρήση PostgreSQL

Στην PostgreSQL, η δημιουργία differential backup έχει αρκετές ομοιότητες με την διαδικασία δημιουργίας incremental, με την διαφορά ότι εδώ θέλουμε την καταγραφή όλων των αλλαγών που έγιναν από το τελευταίο πλήρες backup, επιτυγχάνοντας έτσι ταχύτερη επαναφορά δεδομένων.

Εφόσον έχουμε ενεργοποιήσει το WAL Archiving με τον ίδιο τρόπο όπως είχαμε προαναφέρει και στο [incremental](#) κομμάτι της postgresql , και έχουμε συντάξει την εντολή δημιουργίας πλήρους backup [20] ως εξής:

```
pg_basebackup -D backup -Ft -z -P
```

Εικόνα 2.4.2.1: Σύνταξη εντολής pg_basebackup

, προχωράμε στη δημιουργία differential backup, αντιγράφοντας μόνο τα νέα WAL αρχεία χρησιμοποιώντας την εντολή rsync ως εξής:

```
>rsync -a /var/lib/postgresql/wal_archive/ /var/lib/postgresql/differential_backup/
```

Εικόνα 2.4.2.2: Σύνταξη εντολής rsync

Για την αποκατάσταση των δεδομένων χρησιμοποιούμε το τελευταίο πλήρες backup και τα διαφορικά WAL αρχεία.

Συνοψίζοντας το differential backup μειώνει αρκετά τον χρόνο επαναφοράς δεδομένων, ωστόσο μπορεί να χρειάζεται περισσότερο αποθηκευτικό χώρο λόγω της συσσώρευσης αρχείων.

2.5 SNAPSHOT BACKUP - Αντίγραφο Ασφαλείας Στιγμιότυπου

Το snapshot backup είναι μια μέθοδος δημιουργίας αντιγράφων ασφαλείας που καταγράφει την κατάσταση της εκάστοτε βάσης δεδομένων σε μια συγκεκριμένη χρονική στιγμή. Αντί να αντιγράφονται όλα τα δεδομένα, δημιουργείται ένα στιγμιότυπο του αποθηκευτικού χώρου, καταγράφοντας τις αλλαγές μετά το αρχικό snapshot, επιτυγχάνοντας έτσι ταχύτερη δημιουργία backup και αποκατάσταση των δεδομένων.

Το μεγάλο πλεονέκτημα του αντιγράφου ασφαλείας στιγμιότυπου είναι η αυξημένη ταχύτητα αλλά και αποδοτικότητα, καθώς δεν απαιτεί πλήρη αντιγραφή των δεδομένων. Παρόλα αυτά, οι αποδοτικότητά τους εξαρτάται από το εκάστοτε σύστημα αποθήκευσης και μπορεί να χρειαστεί περισσότερο αποθηκευτικό χώρο αν υπάρχουν πολλές αλλαγές στα δεδομένα.

Τα snapshot backups αποθηκεύει τα μεταδεδομένα που σχετίζονται με κάθε μπλοκ δεδομένων και κάθε φορά που υπάρχει μια αλλαγή, τα νέα μεταδεδομένα καταγράφονται, επιτρέποντας ένα αρχείο καταγραφής αλλαγών και την ανάπτυξη αντιγράφων ασφαλείας σε πραγματικό χρόνο, όταν αποκαλύπτεται ένα σφάλμα ή μια παραβίαση δεδομένων[21].

2.5.1 Δημιουργία snapshot backup με την χρήση MySQL

[22]Προκειμένου να δημιουργήσουμε snapshot backup στη MySQL, χρησιμοποιούμε LVM (Logical Volume Manager) snapshots. Πρέπει ωστόσο να σημειωθεί ότι είναι σημαντικό να κρατηθεί κάποιο μέρος του φυσικού χώρου του δίσκου ως μη διατιθέμενο, καθώς ο μη διατεθειμένος χώρος είναι αυτός που επιτρέπει την δημιουργία snapshot.

Η πιο αποδοτική πρακτική είναι η διάθεση που αποθηκευτικού χώρου που απαιτείται με τα σημερινά δεδομένα συν 10% για ανάπτυξη. Συγκριτικά με τις παραδοσιακές κατατμήσεις δίσκου, η επέκταση ενός τόμου LVM είναι γρηγορότερη, ευκολότερη και δεν απαιτεί χρόνο διακοπής λειτουργίας.

Ωστόσο αρκετοί πάροχοι cloud δεν υποστηρίζουν την τεχνολογία LVM. Παρόλα αυτά για όλους εκείνους τους διακομιστές που μπορούν να χρησιμοποιήσουν LVM, η λειτουργία αντιγράφου ασφαλείας στιγμιότυπου μπορεί να χρησιμοποιηθεί για να ληφθεί ένα εφεδρικό αντίγραφο ασφαλείας MySQL της εκάστοτε βάσης σε συγκεκριμένο χρονικό σημείο με ελάχιστο κλειδώμα πινάκων.

Η χρονική διάρκεια όλης αυτής της διαδικασίας εξαρτάται από την ποσότητα των δεδομένων που πρέπει να εγγραφούν στον δίσκο, αλλά σε γενικές γραμμές είναι πολύ σύντομη.

Με την χρήση της εντολής:

```
FLUSH TABLES WITH READ LOCK
```

Εικόνα 2.5.1.1: Εντολή FLUSH TABLES

κλειδώνουμε όλους τους πίνακες, αποτρέποντας τις εγγραφές και τις αλλαγές στα δεδομένα αλλά επιτρέποντας την ανάγνωση. Εκτελείται πριν την δημιουργία ενός snapshot backup.

Στην συνέχεια δημιουργούμε το snapshot με την χρήση LVM, διαδικασία η οποία είναι σχεδόν στιγμιαία καθώς χρησιμοποιείται μια copy-on-write (COW) snapshot μέθοδος:

```
lvcreate -l100%FREE -s -n mysql-backup /dev/vg0/var
```

Εικόνα 2.5.1.2: Σύνταξη εντολής lvcreate

Τώρα που το snapshot backup έχει επιτυχώς δημιουργηθεί, ξεκλειδώνουμε τους πίνακες με την χρήση της εντολής UNLOCK TABLES.

2.5.2 Δημιουργία snapshot backup με την χρήση PostgreSQL

[23] Προκειμένου να δημιουργήσουμε ένα snapshot backup σε PostgreSQL βάση δεδομένων, χρησιμοποιούμε την εντολή `pg_dump`. Αυτή η εντολή δημιουργεί συνεπή στιγμιότυπα της βάσης, ακόμα και αν αυτή χρησιμοποιείται εκείνη την στιγμή. Η διαφορά της με την MySQL είναι ότι αυτή δεν εμποδίζει άλλους χρήστες να διαβάσουν ή να γράψουν στη βάση.

Ωστόσο πρέπει να υπάρχουν δικαιώματα `SELECT` στην βάση δεδομένων που θέλουμε να δημιουργηθεί το snapshot backup. Η δημιουργία του σε μορφή απλού κειμένου μπορεί να αποκατασταθεί με την χρήση της εντολής `psql`. Τα snapshot backups που δημιουργούνται σε μορφή αρχείου αποκαθίστανται αντίστοιχα με την εντολή `pg_restore`.

Εφόσον επιβεβαιώσουμε ότι το σύστημα αρχείων μας υποστηρίζει τα snapshots, όπως το LVM (Logical Volume Manager) ή το ZFS, εκτελούμε την παρακάτω εντολή:

```
SELECT pg_start_backup('snapshot_label');
```

Εικόνα 2.5.2.1: Σύνταξη εντολής pg_start_backup

, προκειμένου να ενημερώσουμε την PostgreSQL ότι ξεκινά ένα αντίγραφο ασφαλείας, επιτρέποντας την δημιουργία ενός αντιγράφου ασφαλείας στιγμιότυπου.

Στην συνέχεια χρησιμοποιούμε το αντίστοιχο εργαλείο που υποστηρίζει το σύστημα μας για την δημιουργία του στιγμιότυπου. Για παράδειγμα με το LVM εκτελούμε την εντολή:

```
lvcreate --snapshot
```

Εικόνα 2.5.2.2: Εντολή lvcreate

Μετά την επιτυχή δημιουργία του αντιγράφου ασφαλείας στιγμιότυπου, εκτελούμε την εντολή:

```
SELECT pg_stop_backup();
```

Εικόνα 2.5.2.3: Εντολή pg_stop_backup

, ενημερώνοντας έτσι την PostgreSQL για το τέλος του αντιγράφου ασφαλείας και διασφαλίζοντας ότι όλα τα απαραίτητα αρχεία καταγραφής έχουν αποθηκευτεί.

Κεφάλαιο 3ο: Τεχνολογίες

3.1 HTML

[24]Η HTML[44] (HyperText Markup Language) είναι η θεμελιώδης και η πιο διαδεδομένη παγκοσμίως γλώσσα περιγραφής ιδιοτήτων των στοιχείων μιας ιστοσελίδας, που χρησιμοποιείται για την δημιουργία ιστοσελίδων και web εφαρμογών. Χρησιμοποιήθηκε στην πτυχιακή μου εργασία προκειμένου να σχεδιαστεί η διεπαφή του χρήστη, παρέχοντας τη δομή και την βάση για την απόδοση του περιεχομένου της Server Data Backup εφαρμογής μου.

Ο όρος “HyperText” , που μεταφράζεται ως “υπερκείμενο” , αναφέρεται σε συνδέσμους που συνδέουν ιστοσελίδες μεταξύ τους, είτε εντός του ίδιου δικτυακού τόπου είτε μεταξύ δικτυακών τόπων διαφορετικών τοποθεσιών. Οι σύνδεσμοι αυτοί είναι μια θεμελιώδης πτυχή του Παγκόσμιου Ιστού. Με το ανέβασμα οποιουδήποτε είδους περιεχομένου και με την σύνδεση του με ιστοσελίδες που έχουν δημιουργηθεί από άλλους ανθρώπους, γινόμαστε ενεργοί συμμετέχοντες στον Παγκόσμιο Ιστό.

3.1.1 Θετικά χαρακτηριστικά της HTML

[25]Ένα από τα βασικότερα πλεονεκτήματα της HTML είναι η απλότητα της, καθώς επιτρέπει τη γρήγορη ανάπτυξη αλλά και διαμόρφωση ιστοσελίδων. Είναι μια ευρέως υποστηριζόμενη γλώσσα περιγραφής ιδιοτήτων των στοιχείων από τα οποία αποτελείται μια ιστοσελίδα, από όλους τους σύγχρονους browsers, εξασφαλίζοντας την συμβατότητα μεταξύ διαφορετικών πλατφορμών.

Η HTML[44] είναι εύκολη στην εκμάθηση και ο συνδυασμός της με τεχνολογίες JAVASCRIPT[45] και CSS[52], διευκολύνει και βελτιώνει σε τεράστιο βαθμό την εμπειρία χρήστη.

Είναι SEO-Friendly[26], γεγονός που διευκολύνει την ευρετηρίαση των ιστοσελίδων από πολυάριθμες μηχανές αναζήτησης. Αυτό το είδος περιεχομένου έχει δημιουργηθεί με τέτοιο τρόπο που βοηθά τις μηχανές αναζήτησης να το κατατάξουν ψηλά. Αν και λανθασμένα κυκλοφορεί η αντίληψη ότι πρέπει να είναι γεμάτο με λέξεις κλειδιά, στην πραγματικότητα ο στόχος είναι να καθοδηγήσει τις μηχανές αναζήτησης να βρουν, να κατανοήσουν και εν τέλη να συνδέσουν το περιεχόμενο μας με το θέμα που προσπαθούμε να καλύψουμε.

3.1.2 Αρνητικά χαρακτηριστικά της HTML

[27]Παρότι η HTML είναι απαραίτητη για την δομή των ιστοσελίδων, έχει περιορισμένες επιλογές σχεδίασης. Δεν μπορεί από μόνη της να παρέχει προηγμένα στυλιστικά χαρακτηριστικά, καθώς το σχεδιαστικό κομμάτι αναλαμβάνεται κυρίως από την CSS. Χωρίς την χρήση της CSS, οι ιστοσελίδες μπορεί να είναι ανταποκρίσιμες και να εκτελούν τα εκάστοτε αιτήματα του χρήστη αλλά φαίνονται απλές και μη στυλιστικές. Επίσης η HTML δεν υποστηρίζει δυναμικά animations ή αλληλεπιδράσεις, κάτι που απαιτεί ξανά την χρήση άλλης τεχνολογίας, στην περίπτωση αυτή της JAVASCRIPT.

Ο σχεδιασμός με HTML είναι λιγότερο ευέλικτος σε σύγκριση με άλλες τεχνολογίες, όπως τα frameworks (π.χ. React, Angular), που προσφέρουν πιο προσαρμόσιμες λύσεις. Γενικότερα για σύνθετες διεπαφές χρήστη, η HTML πρέπει να συνδυάζεται με άλλες τεχνολογίες, γεγονός που αυξάνει την δυσκολία ανάπτυξης.

Ένα ακόμη σημαντικό μειονέκτημα της HTML είναι η μη υποστήριξη offline περιήγησης. Οι ιστοσελίδες που δημιουργούνται με αυτήν απαιτούν να υπάρχει σύνδεση στο διαδίκτυο προκειμένου να μπορεί να φορτωθεί το περιεχόμενο. Όταν ένας χρήστης προσπαθεί να επισκεφθεί μια HTML σελίδα χωρίς σύνδεση, το πρόγραμμα περιήγησης δεν μπορεί να ανακτήσει αρχεία από τον διακομιστή, με αποτέλεσμα η σελίδα να μην είναι διαθέσιμη. Για την υποστήριξη offline browsing, απαιτείται η χρήση τεχνολογιών όπως το Service Workers με το Cache API.

[28] Οι Service Workers ουσιαστικά είναι σκριπτάκια που τρέχουν στο background ενός browser και επιτρέπουν την αποθήκευση και διαχείριση δεδομένων για offline χρήση. Όταν ένας χρήστης επισκέπτεται ξανά τη σελίδα, ο Service Worker μπορεί να εξυπηρετήσει τα δεδομένα από την μνήμη cache, βελτιώνοντας την ταχύτητα και την διαθεσιμότητα του διαδικτυακού περιεχομένου.

Χωρίς αυτές τις τεχνικές και με την έλλειψη σύνδεσης στο διαδίκτυο, οι HTML σελίδες δεν μπορούν να λειτουργήσουν, καθιστώντας την εμπειρία του χρήστη περιορισμένη σε περιβάλλοντα με χαμηλή ή μηδενική συνδεσιμότητα. Έτσι καταλήγουμε πάλι στο συμπέρασμα ότι η HTML πρέπει να συνδυάζεται με άλλες τεχνολογίες για την παροχή πιο αξιόπιστου και δυναμικού περιεχομένου.

3.1.3 Συμπέρασμα και αιτιολόγηση επιλογής της HTML

Λαμβάνοντας υπόψην όλα τα παραπάνω, η HTML[44] αποδεικνύεται ότι ήταν η καταλληλότερη επιλογή καθώς συνδυάζει απλότητα, ευελιξία και συμβατότητα, χαρακτηριστικά που έκαναν δυνατή την δημιουργία αλλά και την λειτουργικότητα της web εφαρμογής μου. Παρότι απαιτεί την χρήση επιπλέον τεχνολογιών για δυναμικότερο περιεχόμενο, παραμένει απαραίτητη στην δημιουργία κάθε web εφαρμογής.

Ένας ακόμη σημαντικός λόγος που την επέλεξα για την πτυχιακή μου εργασία ήταν επειδή επιτρέπει την εύκολη ενσωμάτωση με τις υπόλοιπες τεχνολογίες που χρησιμοποίησα (PHP, SQL, CSS). Αυτή η ευελιξία της με βοήθησε ιδιαίτερα στο να χτίσω την εφαρμογή μου και να κάνω ευκολότερα προσιτή και περισσότερο εύχρηστη την διεπαφή του χρήστη.

3.2 CSS

Η δεύτερη και εξίσου σημαντική τεχνολογία που χρησιμοποίησα για την βελτίωση της διάταξης αλλά και αισθητικής της web εφαρμογής μου είναι η CSS[52] (Cascading Style Sheets). Πρόκειται για ακόμα μία θεμελιώδη τεχνολογία του Web Development, παρέχοντας πολυάριθμες τεχνικές μορφοποίησης των HTML στοιχείων και διαμόρφωσης responsive σχεδίων που προσαρμόζονται σε διάφορες συσκευές.

[29]Με την χρήση της CSS, εξοικονομείται πολύς χρόνος και δουλειά. Μπορεί να ελέγξει την διάταξη πολλών ιστοσελίδων ταυτόχρονα. Υπάρχουν τρεις τρόποι με τους οποίους μπορεί να προστεθεί CSS σε στοιχεία HTML.

Ο πρώτος τρόπος είναι 'INLINE' , που μεταφράζεται ως 'ΕΝΤΟΣ', όπου το χαρακτηριστικό στυλ χρησιμοποιείται μέσα στα στοιχεία HTML. Ο δεύτερος τρόπος είναι μέσα στο αρχείο html όπου το στοιχείο <style> χρησιμοποιείται στην ενότητα <head>. Ο τρίτος και ο πιο διαδεδομένος τρόπος με τον οποίο χρησιμοποιείται η css είναι σε εξωτερικά αρχεία .css, όπου εκεί αποθηκεύονται όλες οι δαιμορφώσεις των στοιχείων. Ένα εξωτερικό αρχείο .css καθορίζει το στυλ για πολλαπλές σελίδες HTML. Μπορεί με οποιαδήποτε αλλαγή πραγματοποιηθεί μέσα σε αυτό το αρχείο να αλλαχτεί η εμφάνιση ολόκληρου του ιστότοπου. Ωστόσο για να μπορεί να χρησιμοποιηθεί ένα ή περισσότερα εξωτερικά φύλλα στυλ, θα πρέπει να προστεθούν οι αντίστοιχοι σύνδεσμοι προς αυτό ή αυτά μέσα στο στοιχείο της HTML.

3.2.1 Θετικά χαρακτηριστικά της CSS

[30]Ένα από τα μεγαλύτερα πλεονεκτήματα της χρήσης CSS[52] είναι ότι εξοικονομεί πολύτιμο χρόνο, καθώς προσαρμόζει την διάταξη πολλών ιστοσελίδων ταυτόχρονα. Όλα τα εξωτερικά stylesheets αποθηκεύονται σε αρχεία .css, τα οποία μπορούν να συνδεθούν με πολλαπλές σελίδες HTML, διευκολύνοντας σε μεγάλο βαθμό τόσο την συντήρηση όσο και την ενημέρωση του σχεδιασμού.

Η χρήση της CSS συμβάλλει επίσης στην βελτίωση της ταχύτητας με την οποία φορτώνει μια ιστοσελίδα, εφόσον μειώνεται σημαντικά η επανάληψη κώδικα καθώς επιτρέπει την αποθήκευση των διαφόρων στυλ που χρησιμοποιεί στην μνήμη cache του browser.

Παράλληλα υποστηρίζει τη δημιουργία responsive σχεδίων, εξασφαλίζοντας έτσι ότι όλες οι ιστοσελίδες θα εμφανίζονται σωστά σε διαφορετικές συσκευές με διαφορετικές διαστάσεις, όπως υπολογιστές, tablets και κινητά τηλέφωνα.

3.2.2 Αρνητικά χαρακτηριστικά της CSS

[31]Παρά τα πολυάριθμα οφέλη της χρήσης CSS, υπάρχουν αρκετά μειονεκτήματα που μπορεί να επηρεάσουν σε μεγάλο βαθμό την ανάπτυξη των ιστοσελίδων. Ένα από αυτά είναι η ασυνέπεια που υπάρχει ανάμεσα διαφορετικών browser, καθώς οι εκδόσεις της CSS ερμηνεύονται με διαφορετικό τρόπο σε προγράμματα περιήγησης (Chrome, Firefox, Safari κτλ.). Αυτό σημαίνει ότι οι προγραμματιστές θα πρέπει να κάνουν επιπλέον δοκιμές και πολύπλοκες προσαρμογές προκειμένου να επιτευχθεί η εκάστοτε συμβατότητα.

Ένα άλλο σημαντικό μειονέκτημα είναι η πολυπλοκότητα της διαχείρισης CSS αρχείων μεγάλου μεγέθους. Όσο αυξάνεται το μέγεθος ενός project, τόσο αυξάνεται και η δυσκολία και πολυπλοκότητα της συντήρησης και της οργάνωσης του κώδικα, ειδικότερα άμα δεν υιοθετείται η χρήση preprocessor CSS εργαλείων (SASS, LESS). Επιπλέον, η CSS δεν διαθέτει ενσωματωμένο μηχανισμό για μεταβλητές και επαναχρησιμοποιήσιμο κώδικα αντίστοιχα, κάνοντας έτσι την ανάπτυξη πιο απαιτητική και χρονοβόρα.

Τελευταίο και άξιο αναφοράς είναι η διευκρίνιση πως όταν αναφερόμαστε στην CSS, δεν αναφερόμαστε σε γλώσσα προγραμματισμού, κάτι που περιορίζει τις δυνατότητες για δυναμική διαχείριση περιεχομένου. Προκειμένου να αντιμετωπιστεί αυτός ο περιορισμός, χρησιμοποιούνται frameworks (όπως το React) αλλά και η Javascript.

3.3 PHP

Σημαντικός παράγοντας για την ανάπτυξη της web εφαρμογής μου και επίτευξης του ζητουμένου της πτυχιακής μου εργασίας ήταν η χρήση της PHP τεχνολογίας, μία από τις πιο διαδεδομένες γλώσσες προγραμματισμού για δυναμικές ιστοσελίδες. [32]Πρόκειται για μια γλώσσα που εκτελείται στον server και επιτρέπει την διαχείριση δεδομένων, την αλληλεπίδραση με βάσεις δεδομένων(που ήταν και το κύριο ζητούμενο της πτυχιακής) αλλά και την δημιουργία δυναμικού περιεχομένου.

[34]Η HTML[44] μπορεί να ενσωματώσει την PHP[36][41] και με αυτό τον τρόπο καθίστανται δυνατή η επικοινωνία με οποιαδήποτε βάση δεδομένων και η εκτέλεση ποικιλίας εργασιών. Αυτό σημαίνει ότι τόσο η είσοδος χρήστη όσο και άλλες μεταβλητές μπορούν να χρησιμοποιηθούν για την τροποποίηση ιστοσελίδων. Επίσης η php χρησιμοποιείται για αποστολή και λήψη cookies, για συλλογή μορφοποιημένων δεδομένων αλλά και για ενημέρωση πληροφοριών βάσεων δεδομένων.

“PHP: Hypertext Preprocessor” είναι το ακρωνύμιο της PHP. Επειδή αναφερόμαστε σε μια γλώσσα αποτελούμενη από ένα σύνολο ενεργειών που πραγματοποιούνται από την πλευρά του διακομιστή, ο κώδικας εκτελείται στον διακομιστή πριν το πρόγραμμα περιήγησης του χρήστη λάβει το περιεχόμενο της ιστοσελίδας.

3.3.1 Θετικά χαρακτηριστικά της PHP

[34]Ένας από τους κυριότερους λόγους που επέλεξα την php είναι η απλότητα της αλλά και το πόσο εύκολα έμαθα τις τεχνολογίες της και το πόσο βοηθητικές ήταν στη δημιουργία της εφαρμογής μου. Η σύνταξη της είναι παρόμοια με αυτή των γλωσσών προγραμματισμού όπως είναι η C και η Javascript, κάτι που με διευκόλυνε σε μεγάλο βαθμό. Επιπλέον το γεγονός ότι είναι ανοιχτού κώδικα την κάνει πιο προσιτή και πιο οικονομική λύση για να αναπτυχθούν web εφαρμογές.

Σημαντικό πλεονέκτημα της χρήσης της είναι η υποστήριξη πολλαπλών βάσεων δεδομένων, όπως MySQL[36][39] και PostgreSQL[38] (τα 2 είδη βάσεων που χρησιμοποιήθηκαν στην εφαρμογή), κάνοντας ευκολότερη την υλοποίηση της διαχείρισης αντιγράφων ασφαλείας βάσεων δεδομένων. Ο ταχύτερος ρυθμός φόρτωσης της σε σχέση με άλλες γλώσσες προγραμματισμού ακόμα και με σύνδεση σε δίκτυο αργής ταχύτητας την καθιστά εξαιρετική επιλογή.

Βοηθά σημαντικά στην επαναχρησιμοποίηση ενός ισοδύναμου κώδικα, μειώνοντας έτσι την ανάγκη χρήσης μακροσκελή κώδικα και πολύπλοκων δομών του για την εκτέλεση των απαιτούμενων εργασιών της εφαρμογής.

[35]Ταυτόχρονα διαθέτει ενσωματωμένους μηχανισμούς για την ανίχνευση και διαχείριση σφαλμάτων, διευκολύνοντας σημαντικά την επιδιόρθωσή τους. Οι κύριοι τύποι σφαλμάτων είναι τα Parse Errors (Συντακτικά Σφάλματα), τα Fatal Errors (Κρίσιμα Σφάλματα) και τα Warnings & Notices (Προειδοποιήσεις και Ειδοποιήσεις). Με την σωστή διαχείριση σφαλμάτων βελτιώνεται σημαντικά η ασφάλεια και η σταθερότητα των εφαρμογών.

3.3.2 Αρνητικά χαρακτηριστικά της PHP

[36]Παρότι η PHP περικλείεται από αρκετά θετικά χαρακτηριστικά, την συνοδεύουν και μερικά σημαντικά μειονεκτήματα, με το κυριότερο από αυτά να είναι οι κίνδυνοι ασφάλειας.

Λόγω του ότι αναφερόμαστε σε μια ανοιχτού κώδικα τεχνολογία, καθώς και του εύκολα προσβάσιμου αρχείου κειμένου ASCII, οι εφαρμογές PHP είναι αρκετά ευάλωτες σε επιθέσεις τύπου SQL Injections και cross-site scripting (XSS). Αυτό ‘αναγκάζει’ τους προγραμματιστές να εφαρμόζουν τακτικά βέλτιστες πρακτικές ασφάλειας και να ενημερώνουν τις γνώσεις τους σχετικά με τα εκάστοτε μέτρα ασφαλείας της PHP.

Η περιορισμένη υποστήριξη multithreading είναι επίσης ένα μεγάλο μειονέκτημα της PHP, καθώς το μονονηματικό μοντέλο που ακολουθεί η PHP δεν την καθιστά την καλύτερη επιλογή για πιο περίπλοκες multithreading εφαρμογές. Ενώ η χρήση εξωτερικών βιβλιοθηκών και η ενσωμάτωση με άλλες τεχνολογίες περιορίζουν ως ένα βαθμό το πρόβλημα αυτό, αυτόματα προστίθενται αρκετή πολυπλοκότητα στην διαδικασία ανάπτυξης των εκάστοτε εφαρμογών.

[37]Ενώ η PHP έχει πολλές ενημερώσεις, εξακολουθεί να θεωρείται λιγότερο πλούσια σε χαρακτηριστικά σε σύγκριση με νεότερες γλώσσες προγραμματισμού που έχουν σχεδιαστεί για την ανάπτυξη ιστοσελίδων, όπως για παράδειγμα η JavaScript (Node.js) ή η Python.

3.4 SQL

Η πιο δημοφιλής γλώσσα που χρησιμοποίησα για την διαχείριση βάσεων δεδομένων, που στην περίπτωση της εφαρμογής αυτής κύριο ζητούμενο ήταν η δημιουργία αντιγράφων ασφαλείας βάσεων δεδομένων, είναι η SQL (Structured Query Language). Συγκεκριμένα, επέλεξα να χρησιμοποιήσω MySQL έναντι της PostgreSQL λόγω της εξαιρετικής ευκολίας της στην χρήση και της ευχρηστίας της, χαρακτηριστικά που την καθιστούν ιδανικότερη για την ανάπτυξη της web εφαρμογής μου. Παρότι η PostgreSQL είναι πιο προηγμένη, καθώς υποστηρίζει πιο σύνθετα χαρακτηριστικά και τύπους δεδομένων, η MySQL προσφέρει μια πιο απλοποιημένη εμπειρία, κάτι ιδιαίτερα βοηθητικό στην δημιουργία της web εφαρμογής μου. Η υποστήριξη της από πολλές πλατφόρμες και το γεγονός ότι προσφέρει υψηλή απόδοση για εφαρμογές με υψηλό φόρτο αναγνωστικών λειτουργιών, την έκαναν μονόδρομη επιλογή για αυτό το είδος εφαρμογής.

[38]Γενικότερα, σε μια χρονική περίοδο όπου τα δεδομένα είναι κρίσιμο μέρος πολυάριθμων web και mobile εφαρμογών, χρειάζονται βάσεις δεδομένων και ένας τρόπος για τους χρήστες να μπορούν να

διαχειρίζονται αυτά τα δεδομένα. Η SQL είναι η γλώσσα προγραμματισμού που δίνει την δυνατότητα στους προγραμματιστές να διαχειρίζονται αυτά τα δεδομένα.

Ενώ η SQL θεωρείται γλώσσα προγραμματισμού, διαφέρει από τις άλλες γλώσσες προγραμματισμού όπως είναι η C++ και η Java (γλώσσες 3ης γενιάς έναντι της Sql που είναι 4ης). Η σημαντικότερη διαφορά της με τις υπόλοιπες διαδεδομένες γλώσσες προγραμματισμού είναι ότι είναι αρκετά μοναδική, καθώς πρόκειται κυρίως για μια γλώσσα ερωτημάτων που επιτρέπει την αποθήκευση, ανάκτηση αλλά και επεξεργασία δεδομένων σε βάσεις δεδομένων.

3.4.1 Θετικά χαρακτηριστικά της SQL

[39]Ένα από τα κύρια πλεονεκτήματα της SQL είναι η απλότητα των εντολών ανάκτησης και επεξεργασίας δεδομένων βάσεων που χρησιμοποιεί. Με τον συνδυασμό των εντολών SELECT, INSERT, UPDATE και DELETE, δίνεται η δυνατότητα στους προγραμματιστές να ανασύρουν και να επεξεργαστούν συγκεκριμένα σημεία δεδομένων των εκάστοτε βάσεων. Βοηθάνε στην πραγματοποίηση τροποποιήσεων σε υπάρχουσες εγγραφές καθώς και στην αφαίρεση ανεπιθύμητων δεδομένων. Σε μια εποχή που οι επιχειρήσεις υφίστανται ταχεία αύξηση των δεδομένων, η ικανότητα εύκολης και αποτελεσματικής συσχέτισης των δεδομένων καθίστανται ζωτικής σημασίας πτυχή.

Η ασφάλεια των δεδομένων αποτελεί άλλη μια ύψιστης σημασίας προτεραιότητα στον σύγχρονο ψηφιακό κόσμο, με τις απειλές στον κυβερνοχώρο να αυξάνονται και να εξελίσσονται τακτικά. Οι βάσεις δεδομένων SQL παρέχουν ισχυρούς μηχανισμούς εξασφαλίζοντας όσο το δυνατόν μεγαλύτερη προστασία των ευαίσθητων πληροφοριών. Οι τρόποι με τους οποίους οι βάσεις δεδομένων SQL διασφαλίζουν όλα τα δεδομένα είναι με επιλογές κρυπτογράφησης αλλά και με ελέγχους πρόσβασης. Μόνο με το κατάλληλο κλειδί αποκρυπτογράφησης μπορεί ένας χρήστης να αποκωδικοποιήσει τα δεδομένα αυτά, καθώς η διαδικασία της κρυπτογράφησης τα μετατρέπει σε μη αναγνώσιμες μορφές.

Όσον αφορά τους ελέγχους πρόσβασης, κατηγοριοποιεί τους χρήστες και τους χορηγεί διαφορετικά επίπεδα δικαιωμάτων, εξασφαλίζοντας ότι μόνο εξουσιοδοτημένο προσωπικό μπορεί να έχει πρόσβαση στα εκάστοτε δεδομένα.

Με τον συνδυασμό όλων αυτών των μέτρων ασφαλείας, οι επιχειρήσεις αποκτούν την δυνατότητα οικοδόμησης υψηλής εμπιστοσύνης με τους πελάτες τους, διασφαλίζουν σε μεγάλο βαθμό τις πληροφορίες ιδιοκτησίας και συμμορφώνονται ευκολότερα με τους κανονισμούς προστασίας δεδομένων. Με τις επιθέσεις στον κυβερνοχώρο και τις παραβιάσεις των δεδομένων να αυξάνονται σε μεγάλο βαθμό στην σημερινή εποχή, η SQL προσφέρει μια ασφαλή βάση για την διαχείριση ευαίσθητων και προσωπικών δεδομένων.

3.4.2 Αρνητικά χαρακτηριστικά της SQL

Παρά τα πολυάριθμα πλεονεκτήματα και οφέλη της SQL στην προστασία και διαχείριση των δεδομένων, παρουσιάζονται και μερικά σημαντικά μειονεκτήματα. [39]

Ένα βασικό ζήτημα είναι οι περιορισμένες δυνατότητες της για οριζόντια κλιμάκωση. Καθώς είναι συνηθισμένο για τις βάσεις των δεδομένων να κλιμακώνονται κάθετα, υπάρχει η απαίτηση για συνεχή αναβάθμιση του υπάρχοντος διακομιστή με περισσότερους πόρους, κάτι που σε περιβάλλοντα με αυξημένες απαιτήσεις δεδομένων, μπορεί να είναι αρκετά δαπανηρό αλλά και περιοριστικό.

Επιπροσθέτως, οι SQL βάσεις δεδομένων είναι σχεδιασμένες με τέτοιο τρόπο που τις καθιστά λιγότερο ευέλικτες για ημιδομημένα ή μη δομημένα δεδομένα, η χρήση των οποίων συνηθίζεται σε σύγχρονες εφαρμογές, λόγω της χρήσης προκαθορισμένων σχημάτων. Αυτή η ανάγκη της για αυστηρή δομή αυξάνει σημαντικά την δυσκολία προσαρμογής της σε μεταβαλλόμενες απαιτήσεις δεδομένων.

[40]Σε θέματα απόδοσης, οι SQL βάσεις δεδομένων μπορεί να παρουσιάσουν μειωμένη ταχύτητα σε εφαρμογές με τεράστιους όγκους δεδομένων ή με υψηλό αριθμό ταυτόχρονων χρηστών. Η ανάγκη τους για πολύπλοκες συνενώσεις (με την χρήση της εντολής JOIN), σε συνδυασμό με την αυστηρή τήρηση των κανόνων ACID, επιβραδύνουν σημαντικά την απόκριση του συστήματος.

Τέλος ένας ακόμα αρνητικός παράγοντας της χρήσης SQL βάσεων είναι το συνολικό κόστος της ιδιοκτησίας τους. Η συντήρηση και η διαχείριση μιας SQL βάσης δεδομένων ειδικότερα σε περιπτώσεις που το σύνολο των δεδομένων είναι υψηλό, απαιτεί πιο ισχυρά υλικά και πιο προηγμένες υποδομές πληροφορικής.

Ο συνδυασμός όλων των παραπάνω αρνητικών παραγόντων καθιστούν τις SQL βάσεις δεδομένων λιγότερο κατάλληλες για εφαρμογές που απαιτούν υψηλή ευελιξία, κλιμάκωση και απόδοση.

Κεφάλαιο 4ο: Σχεδίαση και Ανάπτυξη του BUDBServer (Backup Database Server)

4.1 Λειτουργικές Απαιτήσεις

Ο στόχος της πτυχιακής αυτής εργασίας μου ήταν να υλοποιηθεί μια εφαρμογή, πλήρως λειτουργική με όσο το δυνατόν πιο user-friendly interface, η οποία θα δίνει την δυνατότητα στους χρήστες να δημιουργούν και να διαχειρίζονται αντίγραφα ασφαλείας για MySQL[36][39] και PostgreSQL[38] βάσεις δεδομένων. Η ανάπτυξη αυτής της εφαρμογής βασίστηκε αρκετά σε συγκεκριμένες απαιτήσεις χρηστών (user stories), τις οποίες έλαβα υπόψη για να διασφαλίσω ότι το τελικό αποτέλεσμα καλύπτει τις ουσιαστικές ανάγκες πραγματικών χρηστών.

Ξεκινώντας, η εφαρμογή μου περιλαμβάνει πλήρες σύστημα αυθεντικοποίησης χρήστη, με δυνατότητα εγγραφής (sign up), σύνδεσης (login), καθώς και επαναφοράς κωδικού πρόσβασης (μέσω φόρμας forgot/reset password). Η είσοδος γίνεται αποκλειστικά μέσω email, αυξάνοντας έτσι την ευχρηστία και την ασφάλεια του συστήματος και εξασφαλίζοντας την μοναδικότητα του username του κάθε χρήστη. Για την αποθήκευση των δεδομένων των χρηστών και των αντίστοιχων συνδέσεων τους χρησιμοποιείται μια βάση δεδομένων με την χρήση τριών πινάκων, στους οποίους στηρίζεται η λειτουργικότητα όλης της εφαρμογής.

Η δημιουργία αντιγράφων ασφαλείας όπως προανέφερα καλύβει τόσο MySQL όσο και PostgreSQL servers. Για την επίτευξη αυτής της απαίτησης χρηστών, χρησιμοποιούνται οι αντίστοιχες εντολές backup κάθε συστήματος, οι οποίες εκτελούνται μέσω PHP στο backend κομμάτι. Υποστηρίζεται το βασικό είδος πλήρους αντιγράφου ασφαλείας (FULL BACKUP) , με τα υπόλοιπα 3 είδη (INCREMENTAL, DIFFERENTIAL, SNAPSHOT) να προορίζονται για μελλοντικές επεκτάσεις τις εφαρμογής καθώς υπάρχει ένας μεγάλος περιορισμός δυνατοτήτων με τα δικαιώματα του κάθε server που αποθηκεύει ο κάθε χρήστης καθώς και η έλλειψη ιδιοκτησίας του για κάθε μια από τους αποθηκευμένους του server.

Η διεπαφή της εφαρμογής έχει σχεδιαστεί με μεγαλύτερη προτεραιότητα την φιλικότητα προς τον χρήστη και για αυτό χρησιμοποιήσα AJAX[51] για την επικοινωνία μεταξύ Frontend και Backend. Μέσω μιας συλλογής από endpoints επιτυγχάνεται μια γρήγορη και ομαλή αλληλεπίδραση χωρίς να χρειάζεται σε κάθε ενέργεια του χρήστη να πραγματοποιούνται περιττές ανανεώσεις των σελίδων. Ο χρήστης μπορεί να δημιουργεί νέες συνδέσεις σε βάσεις δεδομένων στις οποίες έχει πρόσβαση, να επιλέγει για ποιά σύνδεση θέλει να δημιουργήσει αντίγραφο ασφαλείας και τι είδους, καθώς και να αποθηκεύει τα στοιχεία αυθεντικοποίησης στο προσωπικό του προφίλ κρυπτογραφημένα για μεγαλύτερη ασφάλεια.

Μια σημαντική απαίτηση χρήστη ήταν η σωστή οργάνωση των αντιγράφων ασφαλείας. Αυτό επιτεύχθηκε με την χρήση της εντολής **mkdir**[50], όπου κατά την εκτέλεση κάθε διαδικασίας δημιουργίας backup δημιουργείται αυτόματα φάκελος στον server, οργανωμένος ανά χρήστη και ανά σύνδεση βάσης δεδομένων, με το όνομα του οποίου να βασίζεται στο email του χρήστη. Επιπλέον, χάρη σε αυτή την οργάνωση, ο χρήστης μπορεί να προβάλει τα αποθηκευμένα του αντίγραφα ασφαλείας, με πληροφορίες για το πότε δημιουργήθηκε και τι είδους είναι αντίστοιχα. Ο χρήστης μπορεί να κατεβάσει τοπικά κάθε αρχείο που έχει δημιουργήσει αλλά και να το διαγράψει σε περίπτωση που το επιθυμεί.

Συνοψίζοντας, κάθε μία από αυτές τις επιλογές τεχνολογιών και εργαλείων έγινε με κεντρικό γνώμονα τις ανάγκες του κάθε συστήματος και των χρηστών του. Αυτό που είχα κυριότερο στόχο ήταν να

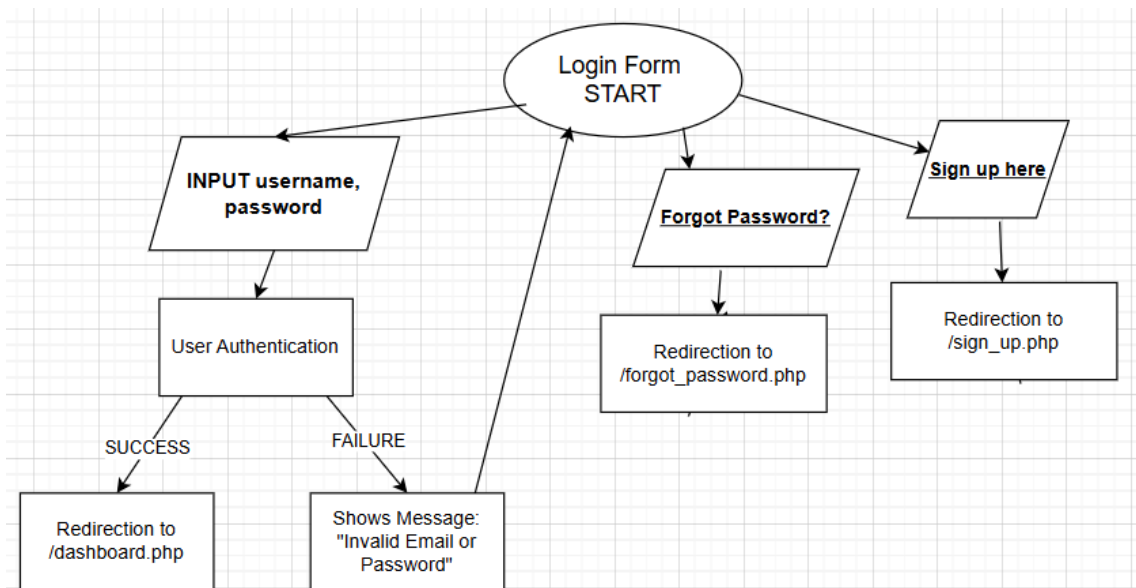
αποδείξω πως με την χρήση της PHP και κανενός framework μπορεί να υπάρχει μια πλήρως λειτουργική εφαρμογή που να παρέχει μια υψηλής σημασίας υπηρεσία, που είναι η διαχείριση αντιγράφων ασφαλείας βάσεων δεδομένων, με πρακτική αξία στον χώρο της διαχείρισης και προστασίας των δεδομένων στην σύγχρονη αυτή εποχή.

4.2 Αρχιτεκτονική Εφαρμογής

4.2.1 Flowcharts (Διαγράμματα Ροής)

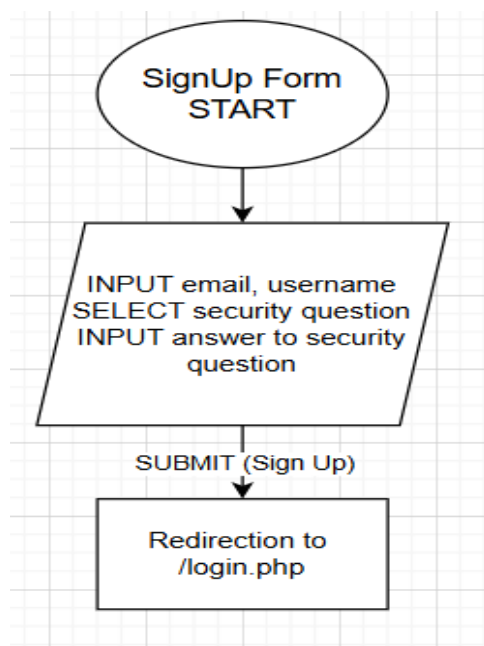
Παρακάτω παρουσιάζονται τα διαγράμματα ροής της εφαρμογής μου, που αναπαριστούν με απλό και κατανοητό τρόπο πως συνδέονται τα πάντα στην εφαρμογή μου και πως λειτουργεί με τον συνδυασμό τους. Μέσω αυτών φαίνεται η ροή των δεδομένων και τα βασικά βήματα που ακολούθησα για να επιτευχθούν όλες οι λειτουργικές απαιτήσεις της εφαρμογής που προανέφερα στην προηγούμενη ενότητα.

LOGIN FORM



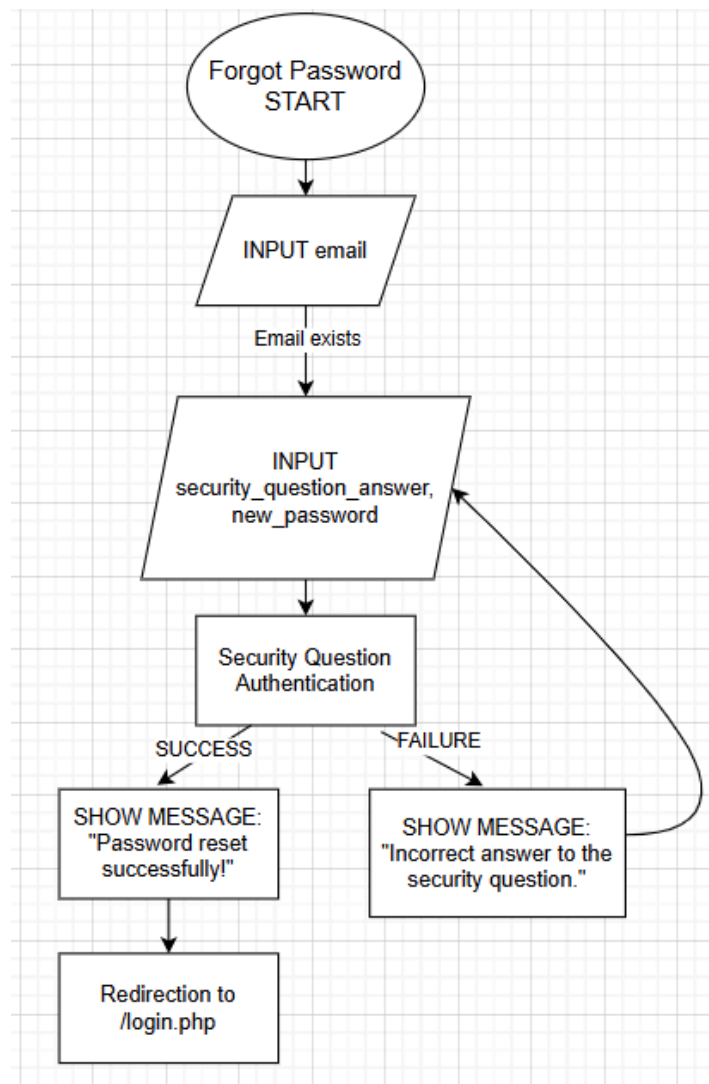
Εικόνα 4.2.1.1: Διάγραμμα ροής LOGIN φόρμας

SIGN-UP FORM



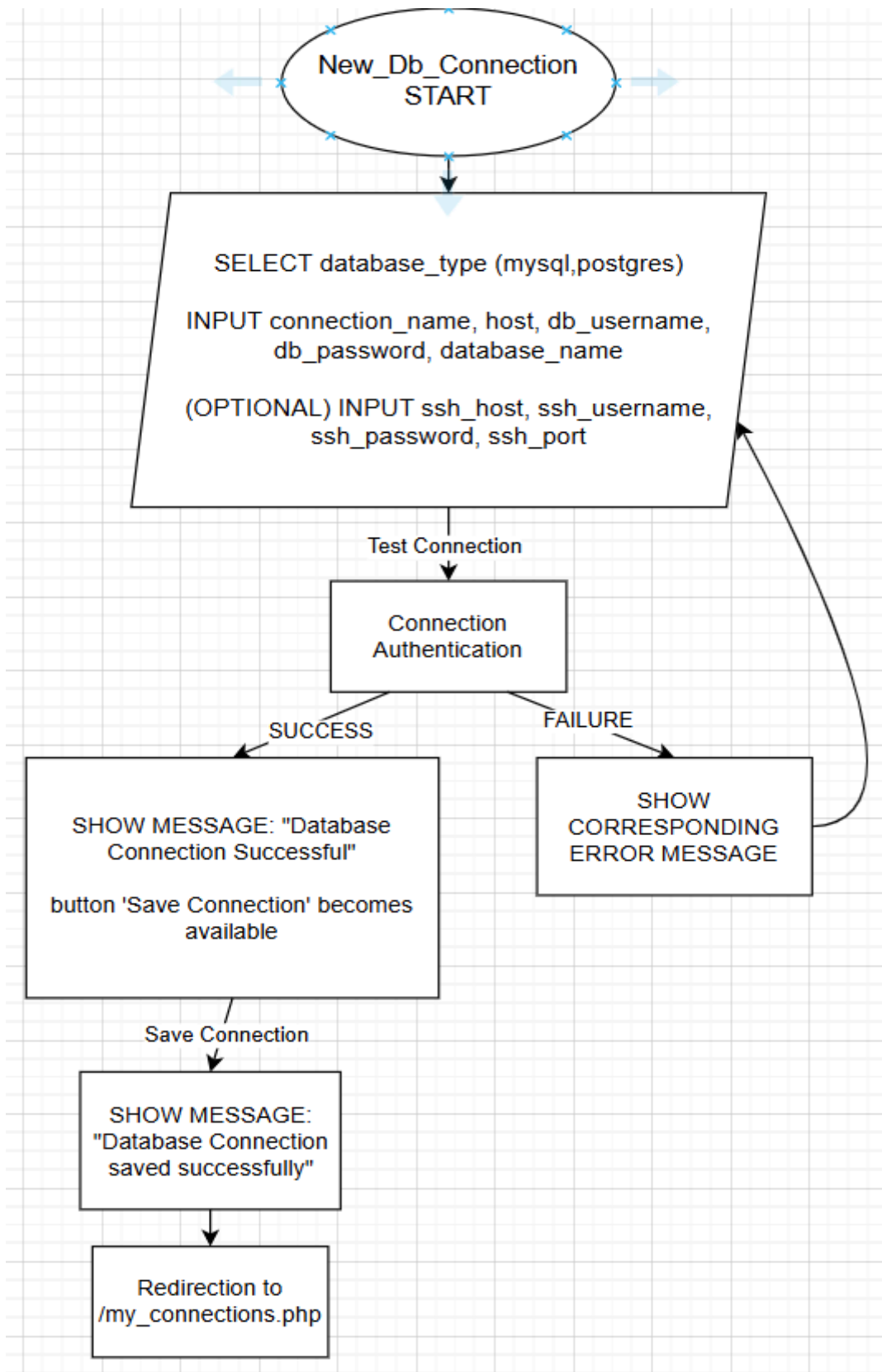
Εικόνα 4.2.1.2: Διάγραμμα ροής SIGN_UP φόρμας

FORGOT PASSWORD



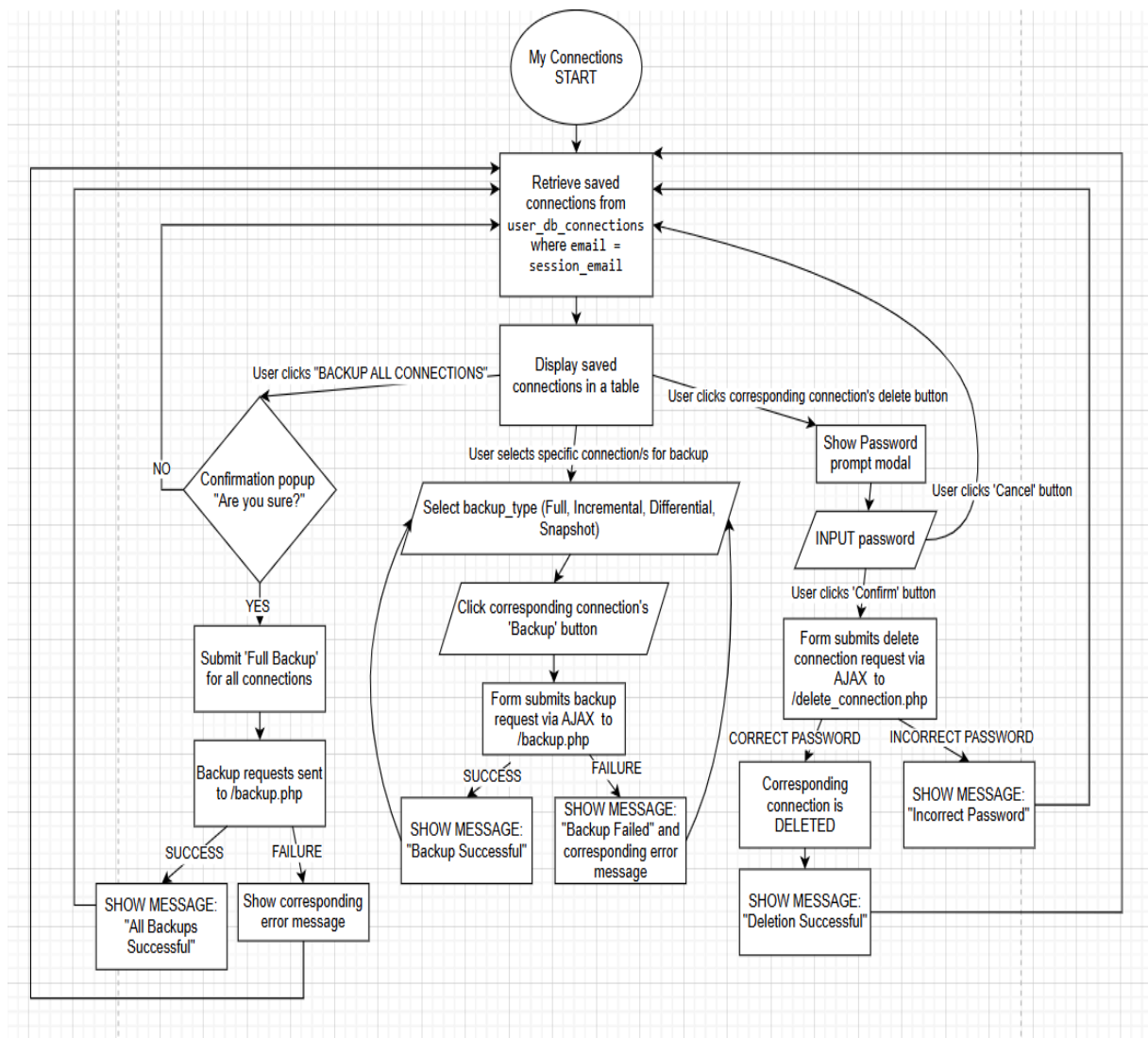
Εικόνα 4.2.1.3: Διάγραμμα ροής FORGOT_PASSWORD φόρμας

NEW DATABASE CONNECTION FORM



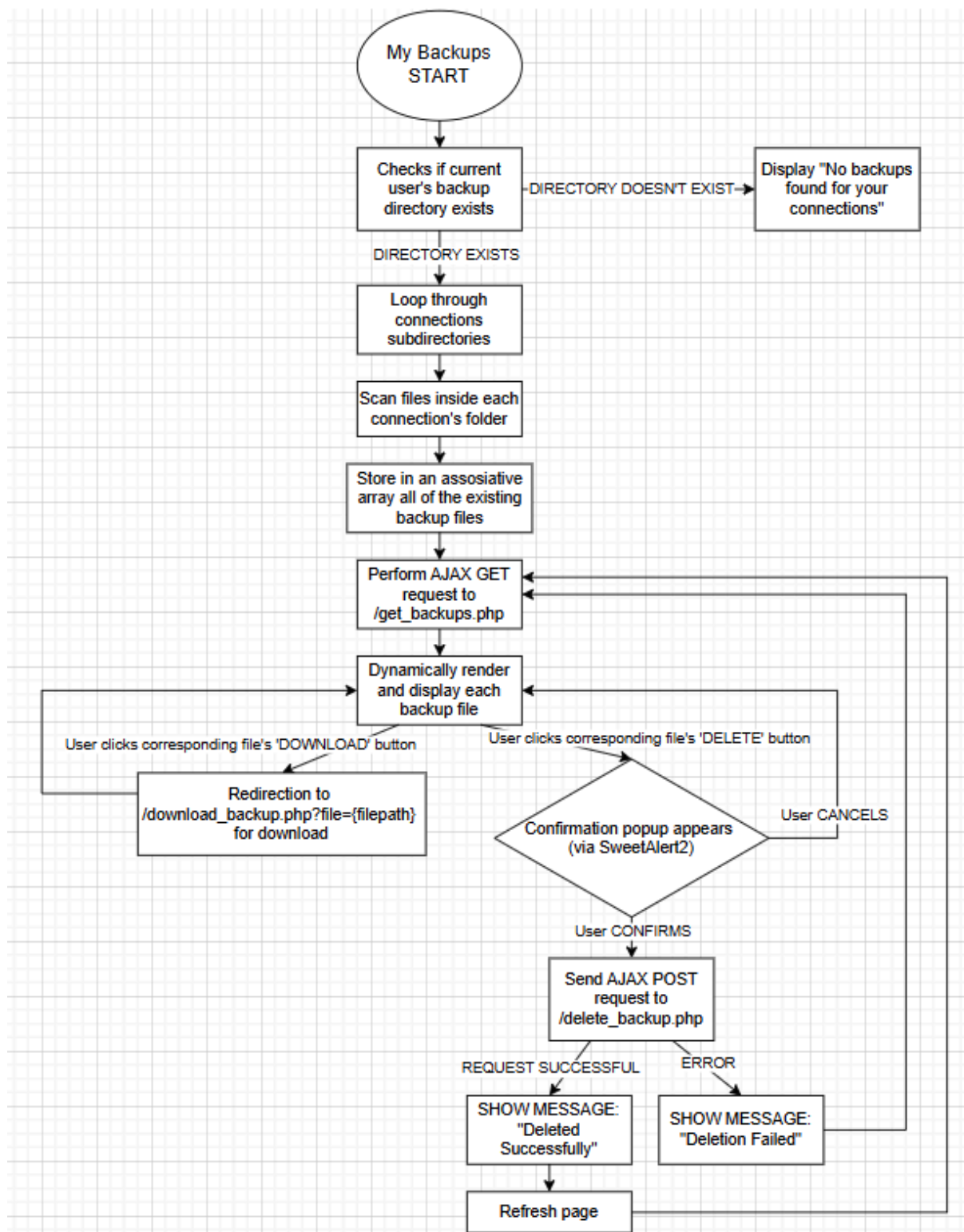
Εικόνα 4.2.1.4: Διάγραμμα ροής NEW DATABASE CONNECTION φόρμας

MY CONNECTIONS



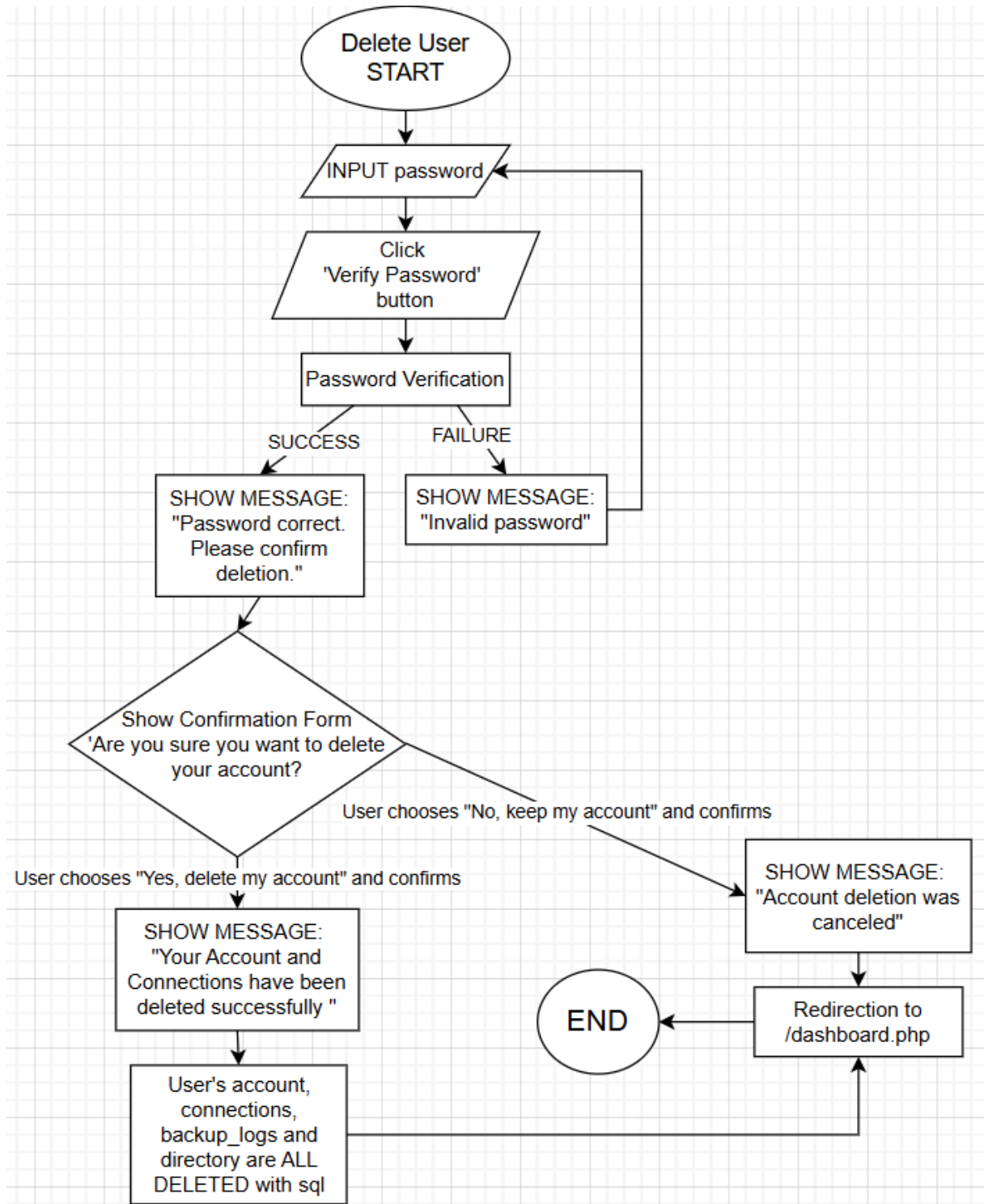
Εικόνα 4.2.1.5: Διάγραμμα ροής MY CONNECTIONS φόρμας

MY BACKUPS



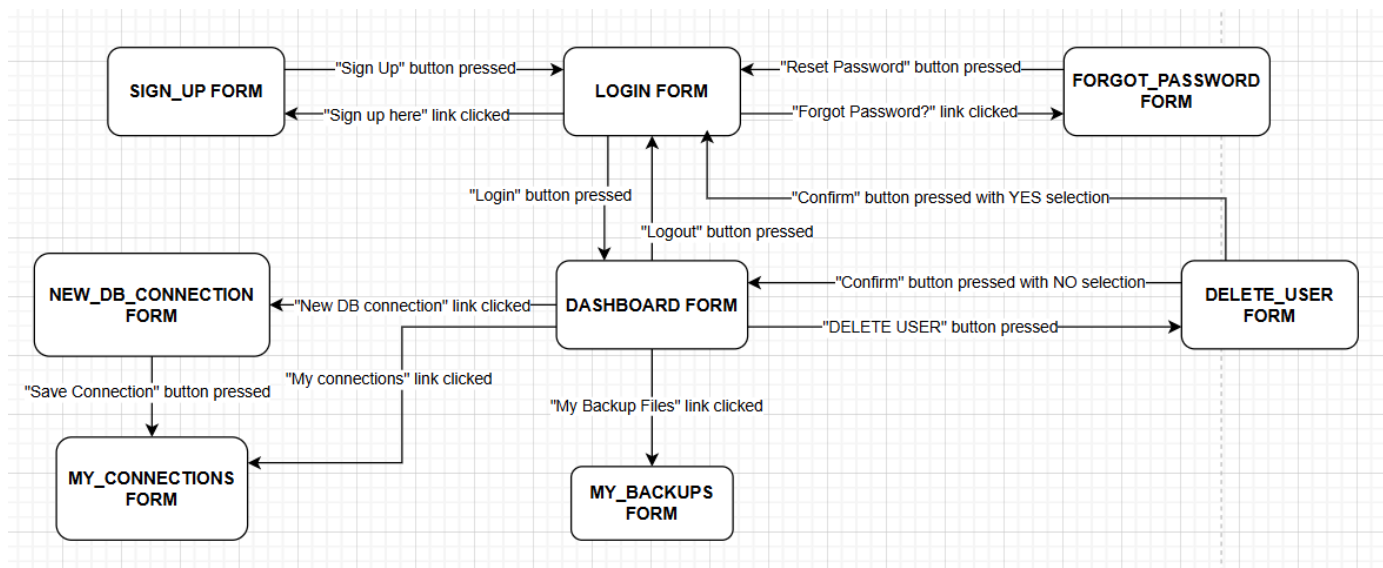
Εικόνα 4.2.1.6: Διάγραμμα ροής MY BACKUPS φόρμας

DELETE USER FORM



Εικόνα 4.2.1.7: Διάγραμμα ροής DELETE USER φόρμας

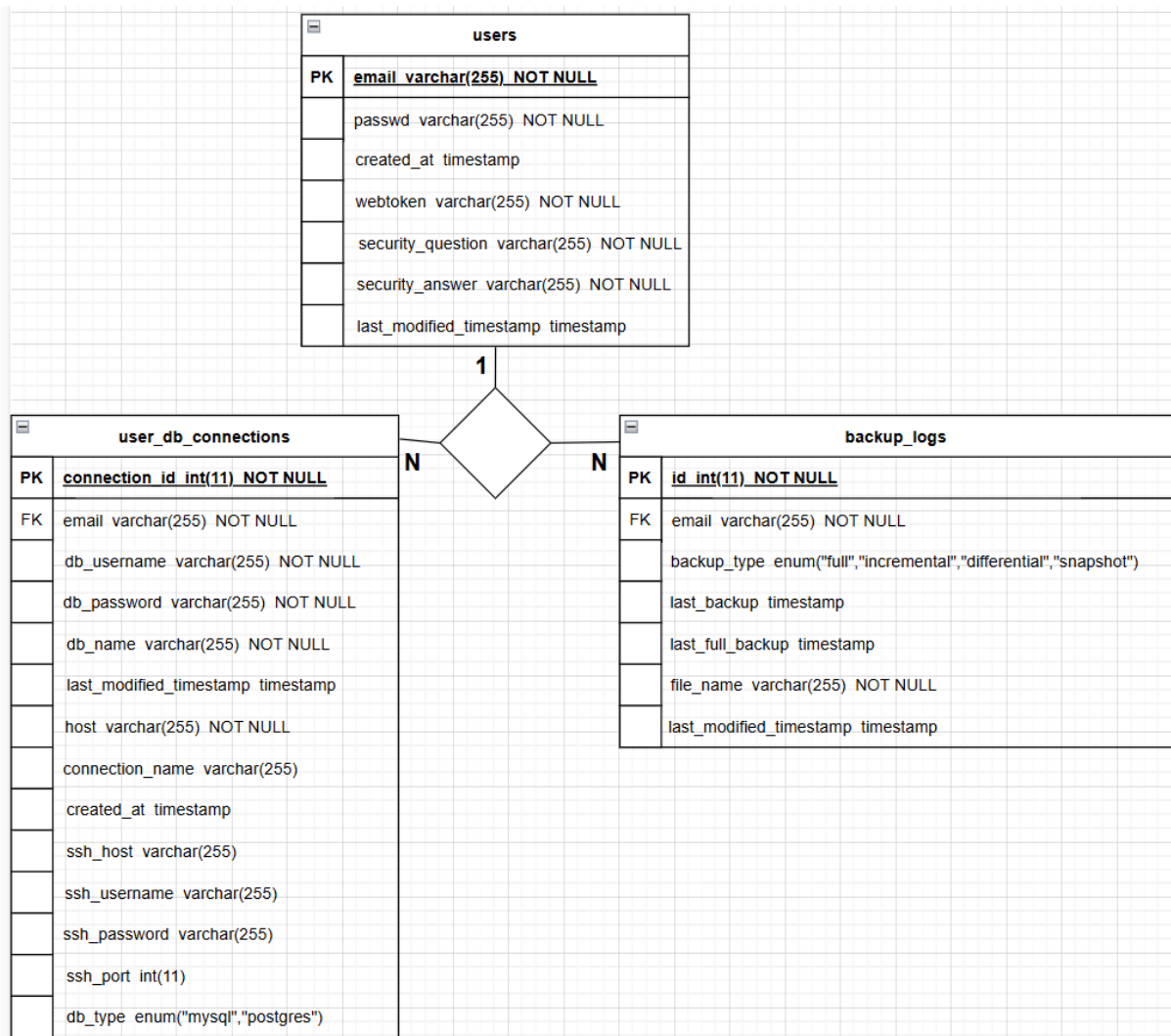
ΓΕΝΙΚΟ ΔΙΑΓΡΑΜΜΑ ΡΟΗΣ ΕΦΑΡΜΟΓΗΣ



Εικόνα 4.2.1.8: Διάγραμμα ροής εφαρμογής

4.2.2 ER Diagram (Διάγραμμα Σχέσης Οντοτήτων)

Η κύρια λειτουργία της πτυχιακής μου εφαρμογής είναι η δημιουργία και η διαχείριση αντιγράφων ασφαλείας βάσεων δεδομένων. Η εφαρμογή αυτή υποστηρίζει πολλούς παρόχους, χρησιμοποιεί SSH Tunneling[48] για ασφαλείς απομακρυσμένες συνδέσεις και αποθηκεύει τις ρυθμίσεις κάθε χρήστη. Παρακάτω παρουσιάζεται το ER διάγραμμα της βάσης δεδομένων που χρησιμοποίησα για την λειτουργικότητα της εφαρμογής.



Εικόνα 4.2.2.1: Διάγραμμα σχέσης οντοτήτων εφαρμογής (ER)

Η εφαρμογή μου χρησιμοποιεί 3 πίνακες. Ο πίνακας **users**, που αποθηκεύει τα στοιχεία αυθεντικοποίησης των εκάστοτε χρηστών, έχει σχέση 1 προς N με τους άλλους 2 πίνακες, εκ των οποίων ο πίνακας **user_db_connections** αποθηκεύει πληροφορίες και στοιχεία αυθεντικοποίησης για τις εκάστοτε συνδέσεις βάσεων δεδομένων του κάθε χρήστη και ο πίνακας **backup_logs** περιέχει καταγραφές δημιουργίας αντιγράφων ασφαλείας ως ιστορικό, καθώς και πληροφορίες για το είδος των εκάστοτε αντιγράφων που δημιουργήθηκαν.

4.3 Υλοποίηση του FRONT END

Η κυριότερη απαίτηση των χρηστών είναι η ομαλή αλληλεπίδραση τους με την εφαρμογή και η εξασφάλιση ότι ανα πάσα χρονική στιγμή θα μπορούν να δημιουργούν ψηφιακά αντίγραφα ασφαλείας. Λαμβάνοντας υπόψη αυτό, πρωταρχικός μου στόχος ήταν να δημιουργήσω ένα περιβάλλον φιλικό προς τον χρήστη, εύκολα κατανοητό, όσο το δυνατόν πιο ευχάριστο αισθητικά και με εξασφαλισμένη την λειτουργικότητα του. Για να επιτευχθεί αυτό, χρησιμοποίησα διάφορες web τεχνολογίες και εργαλεία.

Η βασική δομή των σελίδων δημιουργήθηκε με **HTML5**, ενώ στο κομμάτι της αισθητικής και της διαμόρφωσης της διεπαφής των χρηστών χρησιμοποιήθηκε η **CSS3**. Ιδιαίτερη προσοχή έδωσα στο responsive design και στην σωστή οργάνωση των στοιχείων που εμφανίζονται στις σελίδες, προκειμένου να διευκολύνεται όσο το δυνατόν περισσότερο η περιήγηση του χρήστη στην εφαρμογή.

4.3.1 Κώδικας CSS

```
*{
    margin: 0;
    padding: 0;
    box-sizing: border-box;
    font-family: "Poppins", sans-serif;
}
```

Σύνταξη layout με την χρήση του universal selector

Αρχικά, το βασικότερο κομμάτι της εμφάνισης της εφαρμογής στηρίζεται στην χρήση του * (universal selector). Με τον μηδενισμό του padding και την εφαρμογή του box-sizing με την τιμή border-box, επιτυγχάνεται η συνεκτικότητα του layout της εφαρμογής ανάμεσα σε διαφορετικούς browsers, δίνοντας έτσι την δυνατότητα η εφαρμογή να εμφανίζεται σωστά σε οποιονδήποτε browser χρησιμοποιεί ο εκάστοτε χρήστης.

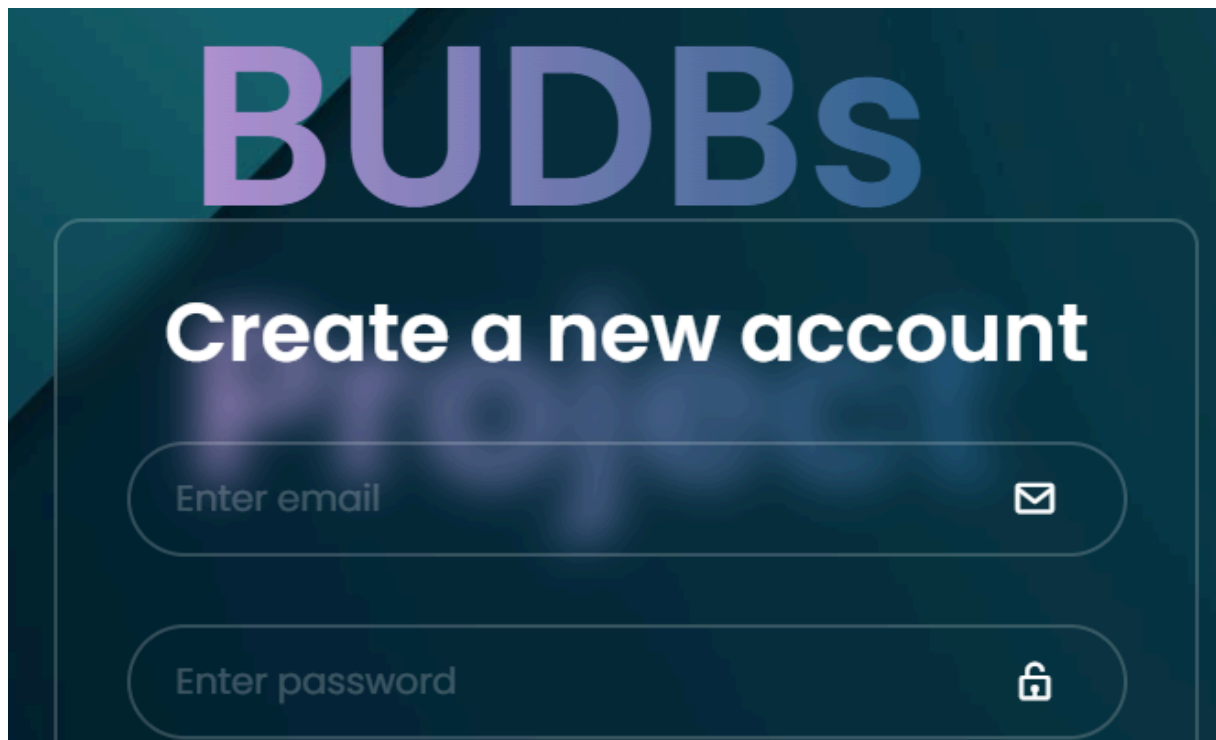
Κεφάλαιο 3

Μεγάλη σημασία έχει και η εμφάνιση του φόντου της εφαρμογής, το οποίο χρησιμοποιείται σε όλες τις φόρμες της.

```
html, body{  
    display: flex;  
    justify-content: center;  
    align-items: center;  
    min-height: 100vh;  
    background-image: url(login_background.jpg);  
    background-size: cover;  
    background-position: center;  
    margin:0;  
    width: 100%;  
}
```

Στήσιμο εμφάνισης σώματος HTML

Η χρήση της επιλεγμένης μου εικόνας φόντου με το **background-size : cover** και το **background-position : center**, εξασφαλίζω ότι η εικόνα καλύπτει σωστά κάθε διαφορετικό μέγεθος οθόνης. Για το layout προκειμένου το περιεχόμενο να κεντράρεται τόσο οριζόντια όσο και κάθετα, χρησιμοποιήθηκαν οι ιδιότητες **justify-content: center** και **align-items: center**.



Εικόνα 4.3.1: Εφέ Θαμπώματος

Το κομμάτι που πιάνει ο πίνακας της φόρμας, παρατηρούμε ότι θαμπώνει τον τίτλο της εφαρμογής στο σημείο που καλύβει. Χρησιμοποίησα το **backdrop-filter: blur(10px)** στην κλάση του **.wrapper** προκειμένου να δημιουργήσω το 'εφέ' του θαμπώματος ,για να κάνω πιο όμορφο το design της εφαρμογής και να φαίνεται πιο σύγχρονο το UI.

```
.wrapper {  
  width: 500px;  
  background: transparent;  
  border: 2px solid rgba(255, 255, 255, .2);  
  backdrop-filter: blur(10px);  
  color: white;  
  border-radius: 12px;  
  padding: 30px;  
}
```

Σύνταξη κώδικα ιδιοτήτων εμφάνισης κλάσης .wrapper

Κεφάλαιο 3

Στο θέμα της εμφάνισης των buttons, χρησιμοποίησα τις εξής ιδιότητες στην κλάση **.btn** :

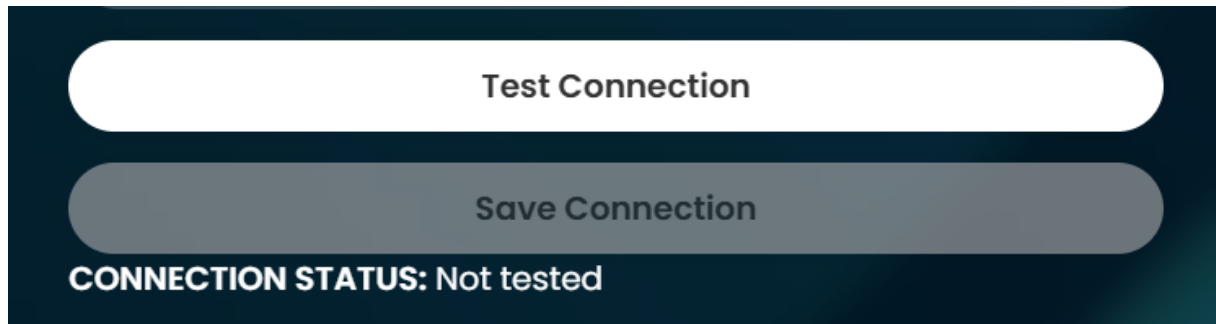
```
.wrapper .btn{  
    width: 100%;  
    height: 45px;  
    background: #fff;  
    border: none;  
    outline: none;  
    border-radius: 40px;  
    box-shadow: 0 0 10px rgba(0, 0, 0, .1);  
    cursor: pointer;  
    font-size: 16px;  
    color: #333;  
    font-weight: 600;  
    margin-top: 15px;  
}
```

Ιδιότητες εμφάνισης κουμπιών στην κλάση .btn

```
.btn:disabled {  
    background-color: #b0b0b0;  
    cursor: not-allowed;  
    opacity: 0.6;  
}
```

Ιδιότητες εμφάνισης disabled button

και το αποτέλεσμα είναι τα κουμπιά να εμφανίζονται ως εξής:



Εικόνα 4.3.2: Εμφάνιση enabled και disabled buttons

Με τον συνδυασμό των παραπάνω ιδιοτήτων, κατάφερα να εμφανίσω τα κουμπιά με στρογγυλεμένες γωνίες, διακριτική σκιά για βάθος αλλά και ευδιάκριτα hover states, βελτιώνοντας τόσο την εμφάνιση τους στην εφαρμογή, όσο και την αλληλεπίδραση του χρήστη με αυτή. Η διαχείριση της κατάστασης “**disabled**” (απενεργοποιημένο κουμπί) με το αλλαγμένο χρώμα και την μειωμένη διαφάνεια βελτιώνει σημαντικά την αισθητική της εφαρμογής και την λειτουργικότητα της.

Κεφάλαιο 3

Όσον αφορά την εμφάνιση των select boxes, χρησιμοποίησα έναν συνδυασμό hover και focus ιδιοτήτων, δίνοντας τα μια ξεχωριστή και σύγχρονη εμφάνιση, κάνοντας τα ελκυστικά και ευχάριστα στην χρήση τους, συνδυάζοντας εμφάνιση με λειτουργικότητα.

```
.select-box {  
    position: relative;  
    width: 100%;  
    margin-bottom: 1rem;  
    margin-right: 12%;  
}  
  
.select-box select {  
    width: 100%;  
    padding: 0.75rem;  
    font-size: 1rem;  
    border: 1px solid #ccc;  
    border-radius: 5px;  
    background-color: #f8f8f8;  
    color: #333;  
    appearance: none;  
    font-family: 'Poppins', sans-serif;  
    outline: none;  
    transition: all 0.3s ease;  
}  
  
.select-box:hover select {  
    border-color: #666;  
    background-color: #e8e8e8;  
}
```

```
.select-box::after {
    content: '\25BC';
    font-size: 1rem;
    position: absolute;
    top: 50%;
    right: 1rem;
    transform: translateY(-50%);
    pointer-events: none;
    color: #666;
}

.select-box select:focus {
    border-color: #666;
    box-shadow: 0 0 5px rgba(100, 100, 100, 0.3);
}
```

Ιδιότητες εμφάνισης των select boxes

Εικόνα 4.3.3: Select Box φόρμας Sign Up

Τέλος, στο κομμάτι της εμφάνισης χρησιμοποιήθηκαν και *media queries*.

```
@media (max-width: 768px) {  
  
  .input-box {  
  
    flex: 1 1 100%;  
  
  }  
  
}
```

CSS Media Queries

Τα πεδία εισαγωγής γίνονται πιο ευέλικτα και ο σχεδιασμός τους προσαρμόζεται σε μικρότερες οθόνες, εξασφαλίζοντας μια πιο φιλική εμπειρία χρήστη σε κάθε συσκευή με μικρότερη οθόνη.

4.3.2 Κώδικας Javascript

Στο κομμάτι της αλληλεπίδρασης του χρήστη με την εφαρμογή, χρησιμοποίησα **JAVASCRIPT**[45]. Με την χρήση αυτού κατάφερα να υλοποιήσω δυναμικές λειτουργίες, όπως την εμφάνιση/απόκρυψη στοιχείων, την επιβεβαίωση πριν την διαγραφή ενός αντιγράφου ή χρήστη αντίστοιχα, καθώς και οποιαδήποτε ενέργεια πραγματοποιείται, χωρίς να απαιτούνται περιττές επαναφορτώσεις των σελίδων.

Ένα από τα σημαντικότερα σημεία της εφαρμογής είναι η επιβεβαίωση που ζητείται από τον χρήστη πριν διαγράψει ένα αντίγραφο ασφαλείας.

```
$(document).on('click', '.delete-btn', function () {  
  
  const filePath = $(this).data('file');  
  
  Swal.fire({  
  
    title: 'Are you sure?',  
    text: "This will permanently delete the backup.",  
    icon: 'warning',  
    showCancelButton: true,  
    confirmButtonColor: '#3085d6',  
    cancelButtonColor: '#d33',  
    confirmButtonText: 'Yes, delete it!'  
  
  }).then((result) => {
```

```

        if (result.isConfirmed) {

            $.ajax({

                url: 'delete_backup.php',

                type: 'POST',

                data: JSON.stringify({ file: filePath }),

                contentType: 'application/json',

                success: function (response) {

                    Swal.fire('Deleted!', response.message,
'success').then(() => location.reload());

                },

                error: function (xhr, status, error) {

                    Swal.fire('Error', 'An unexpected error
occurred: ' + error, 'error');

                }

            });

        }

    });

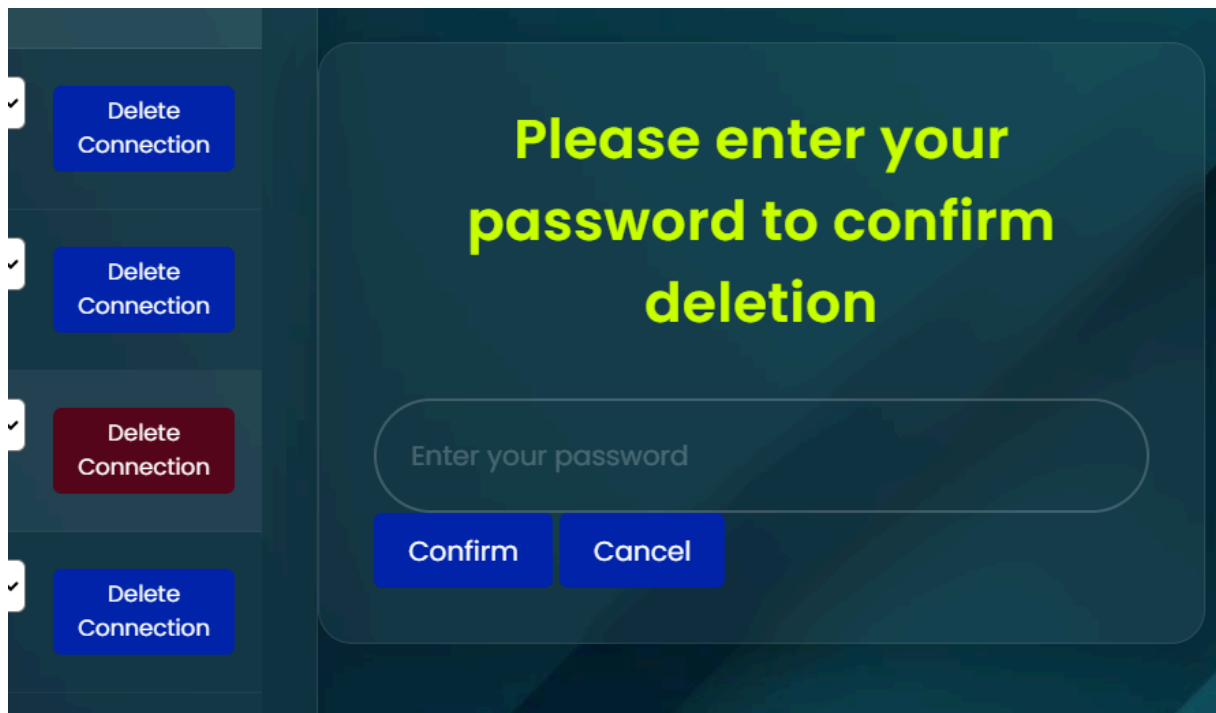
});

```

Χρήση Javascript για δυναμική επιβεβαίωση διαγραφής

Χρησιμοποιούνται **JQuery**[55] και **SweetAlert**[54] τεχνολογίες, προκειμένου να πραγματοποιούνται οι διαγραφές των αρχείων. Όταν ο χρήστης κάνει κλικ σε ένα κουμπί που ανήκει στην κλάση **.delete-btn**, τότε εμφανίζεται ένα προειδοποιητικό μήνυμα με την ερώτηση αν θέλει να προχωρήσει με την διαγραφή. Αν επιβεβαιώσει, τότε γίνεται ένα **AJAX POST** αίτημα στο **delete_backup.php**, στέλνοντας το **filePath** του εκάστοτε αρχείου σε **JSON** μορφή. Αν η διαγραφή του είναι επιτυχής, τότε εμφανίζεται μήνυμα επιτυχίας και η σελίδα ανανεώνεται, ενώ σε περίπτωση σφάλματος εμφανίζεται σχετικό μήνυμα λάθους. Όλη η διαδικασία είναι ασύγχρονη και δεν απαιτείται reload της σελίδας παρα μόνον αν ολοκληρωθεί επιτυχώς η διαγραφή.

Στην συνέχεια, ένα ακόμη χαρακτηριστικό όφελος της χρήσης javascript ήταν η δυνατότητα εμφάνισης δυναμικού περιεχομένου φόρμας εισαγωγής κωδικού για επιβεβαίωση διαγραφής σύνδεσης του εκάστοτε χρήστη, χωρίς να απαιτείται επαναφόρτωση της σελίδας.



Εικόνα 4.3.2.1: Φόρμα υποβολής κωδικού για διαγραφή σύνδεσης

Όταν ο χρήστης πατήσει το κουμπί Delete Connection της αντίστοιχης σύνδεσης που επιθυμεί να διαγράψει, η Javascript παίρνει το **connection_id** της σύνδεσης που πρόκειται να διαγραφεί (**data-connection-id**), αποθηκεύει αυτό το id στο modal (**\$('#password-modal').data('connectionId', ...)**) και εμφανίζει την modal φόρμα (**show()**), η οποία προηγουμένως ήταν κρυφή με την ιδιότητα **display: none**.

```
$("#password-form").on("submit", function(event) {

    event.preventDefault();

    var password = $("input[name='password']").val();

    var connectionId =
    $('#password-modal').data('connectionId');

    $.ajax({

        type: "POST",

        url: "delete_connection.php",

        data: { password: password, connection_id: connectionId

    },
```

```

        dataType: "json",

        success: function(response) {

            if (response.success) {

                Swal.fire('Deleted!', response.message,
'success').then(() => location.reload());

            } else {

                $("#password-error").show();

            }

        }

    });

});

```

Σύνταξη εντολών Javascript για εμφάνιση modal και διαγραφή σύνδεσης

Με το που κάνει ο χρήστης υποβολή του κωδικού στην φόρμα που εμφανίζεται, η js παίρνει τον κωδικό που πληκτρολόγησε, παίρνει το connectionId από το modal, στέλνει AJAX αίτημα στην PHP (delete_connection.php) με δεδομένα τον κωδικό και το connection id και ελέγχει το response. Αν η μεταβλητή success έχει τιμή true, τότε εμφανίζει μήνυμα επιτυχίας και κάνει reload προκειμένου να αφαιρεθεί η διεγραμμένη σύνδεση από τον πίνακα, ενώ αν η τιμή της είναι false, τότε εμφανίζει μήνυμα λάθους για λάθος εισαγωγή κωδικού (**#password-error**) και δεν προχωρά σε διαγραφή.

Στην περίπτωση που ο χρήστης αποφασίσει να ακυρώσει την διαγραφή της σύνδεσης και επιλέξει το κουμπί cancel, τότε καλείται η function **closePasswordModal()**, η οποία κρύβει το modal του κωδικού, κρύβει το μήνυμα λάθους αν αυτό είχε εμφανιστεί και κάνει reset την φόρμα ώστε να καθαριστεί το πεδίο του κωδικού.

```

function closePasswordModal() {

    $('#password-modal').hide(); // κρύβω την φόρμα εισαγωγής κωδικού

    $('#password-error').hide(); // κρυβω τα μηνυματα λαθους

    $('#password-form')[0].reset(); // Reset πεδίου κωδικού

}

```

Σύνταξη εντολών Javascript για απόκρυψη modal

4.3.3 Τεχνολογία AJAX

Ένα ακόμη βασικό κομμάτι του frontend ήταν η ενσωμάτωση της τεχνολογίας **AJAX**[51], η οποία δίνει την δυνατότητα αποστολής αιτημάτων προς το backend με ασύγχρονο τρόπο. Αυτό σημαίνει ότι ο χρήστης μπορούσε να δημιουργήσει αντίγραφα ασφαλείας, να δει τις αποθηκευμένες του συνδέσεις ή να διαγράψει αρχεία χωρίς να “παγώνει” η σελίδα.

```
<script>

    $('#loginForm').on('submit', function(e) {
        e.preventDefault();

        const formData = {
            email: $("input[name='email']").val(),
            password: $("input[name='password']").val(),
        };

        $.ajax({
            type: 'POST',
            url: 'login.php',
            data: formData,
            dataType: 'json',
            success: function(response) {
                console.log('Response:', response);
                if (response.success) {
                    window.location.href = 'dashboard.php';
                } else {
                    alert(response.message);
                }
            },
            error: function(jqXHR, textStatus, errorThrown) {
                console.error('AJAX Error:', textStatus, errorThrown);
                alert('Error logging in: ' + textStatus);
            }
        });
    });

</script>
```

Εικόνα 4.3.1: Χρήση AJAX με JAVASCRIPT

Όπως παρατηρούμε στην εικόνα 4.3.1 , για την υλοποίηση της φόρμας σύνδεσης συνδύασα τις τεχνολογίες Javascript[45] και Ajax[51] , προκειμένου να επιτευχθεί ασύγχρονη επικοινωνία με τον server, χωρίς να απαιτείται ανανέωση της σελίδας. Έτσι βελτιώνεται σημαντικά η εμπειρία χρήστη, καθώς επιτυγχάνεται ομαλότερη ροή και άμεση απόκριση στην πλοήγηση της εφαρμογής.

Αντίστοιχα δουλεύει και η υποβολή του κωδικού του χρήστη, όταν επιλέγει να διαγράψει μια σύνδεση ο χρήστης.

```
$.ajax({
    type: "POST",
    url: "delete_connection.php",
    data: { password: password, connection_id: connectionId
},
    dataType: "json",
    success: function(response) {
        if (response.success) {
            Swal.fire('Deleted!', response.message,
'success').then(() => location.reload());
        } else {
            $("#password-error").show();
        }
    }
});
```

Χρήση AJAX για ασύγχρονη υποβολή κωδικού και confirm

Όταν ο χρήστης υποβάλει τον κωδικό του και πατάει το κουμπί ‘**Confirm**’ , χρησιμοποιείται το AJAX για να σταλούν τα δεδομένα (**password** και **connection_id**) στον server μέσω Javascript, με αποτέλεσμα η σελίδα να μην ανανεώνεται και ο χρήστης να βλέπει το αποτέλεσμα άμεσα.

Ο server απαντάει με JSON δεδομένα που περιέχουν είτε ‘**success:true**’ είτε ‘**success:false**’. Το Javascript χειρίζεται ανάλογα την απόκριση αυτή. Αν είναι επιτυχής, τότε ο χρήστης βλέπει ειδοποίηση επιτυχίας (με SweetAlert[54]) και μετά ανανεώνεται η σελίδα για να ενημερωθεί ο πίνακας, ενώ σε περίπτωση αποτυχίας, εμφανίζεται μήνυμα λάθους χωρίς να ανανεωθεί η σελίδα, κάτι που είναι πολύ πιο πρακτικό και φιλικό προς τον χρήστη.

Με την χρήση AJAX[51] επιτυγχάνεται άμεση δυναμική εμπειρία χρήστη (UX) , καθώς όλη η διαδικασία γίνεται απλή, με τον χρήστη να βλέπει το modal popup, να εισάγει τον κωδικό, να πατάει επιβεβαίωση και να λαμβάνει άμεσα feedback επιτυχίας ή λάθους κωδικού, χωρίς κάποια περιττή καθυστέρηση. Αυτό βελτιώνει περισσότερο την εμπειρία του χρήστη σε σύγκριση με την χρήση κανονικών HTML forms που ανανεώνουν όλη τη σελίδα.

4.4 Συμπεράσματα Front End

Ο συνδυασμός όλων των παραπάνω τεχνολογιών οδήγησε στην επίτευξη μιας ομαλής και ευχάριστης εμπειρίας κατά την χρήση της εφαρμογής, με όλες τις λειτουργίες της να πραγματοποιούνται επιτυχώς και χωρίς καμία περιττή καθυστέρηση. Το frontend κομμάτι χτίστηκε έτσι ώστε η εφαρμογή να φαίνεται επαγγελματική, συνδυάζοντας ομαλή περιήγηση και υψηλή λειτουργικότητα, με κυριότερα θεμέλια της τον συνδυασμό των απαιτήσεων και των αναγκών των χρηστών.

4.5 Υλοποίηση του BACK END

Το backend κομμάτι της εφαρμογής αποτελεί τον κεντρικό άξονα της δομής της εφαρμογής, καθώς σε αυτό υλοποίησα όλες τις λειτουργίες του χρήστη, τις συνδέσεις με τις βάσεις δεδομένων καθώς και την εκτέλεση των εντολών δημιουργίας αντιγράφων ασφαλείας. Η κύρια γλώσσα που χρησιμοποίησα για την ανάπτυξη του backend ήταν η **PHP**, λόγω της σταθερότητας, της ευρείας υποστήριξης αλλά και της δυνατότητας της να ενσωματώνεται εύκολα με HTML[44] και MySQL[36][39].

Χρησιμοποίησα την PHP[36][41] για την υλοποίηση των λειτουργιών της εγγραφής, της σύνδεσης και της επαναφοράς του κωδικού χρήστη, της αποθήκευσης των στοιχείων των συνδέσεων αλλά και για την εκτέλεση των εντολών δημιουργίας αντιγράφων ασφαλείας βάσεων δεδομένων. Για να δημιουργηθούν τα απαραίτητα αρχεία backup τόσο για MySQL όσο και για PostgreSQL, χρησιμοποίησα τα εργαλεία **mysqldump** και **pg_dump** για κάθε βάση αντίστοιχα, τα οποία εκτελούνται μέσω της PHP.

Για την οργάνωση των φακέλων του κάθε χρήστη, χρησιμοποίησα την εντολή **mkdir**[50], προκειμένου να δημιουργείται μια δομή αρχείων βασισμένη όπως προανέφερα στο email του κάθε χρήστη και στο όνομα της εκάστοτε σύνδεσης του. Έτσι, κάθε αρχείο backup αποθηκεύεται ταξινομημένα, επιτυγχάνοντας έτσι πιο ξεκάθαρη και αποτελεσματική διαχείριση τους.

Η βάση δεδομένων της ίδιας της εφαρμογής υλοποιήθηκε με την MySQL, με τα εργαλεία της οποίας οργανώθηκαν οι πίνακες των χρηστών, των συνδέσεων και των ιστορικών των αντιγράφων. Ο λόγος που επέλεξα την MySQL έναντι της PostgreSQL στηρίζεται στην μεγαλύτερη ευκολία στην ρύθμιση της και κυρίως στην μεγάλη υποστήριξη της από την PHP. Είναι πιο διαδεδομένη σε κοινά web hosting περιβάλλοντα, γεγονός που με διευκόλυνε ιδιαίτερα στο στήσιμο της και στην ικανοποίηση των λειτουργικών απαιτήσεων της εφαρμογής μου.

Η ασφάλεια των δεδομένων, ύψιστης σημασίας παράγοντας, διασφαλίστηκε μέσω την χρήσης συναρτήσεων **password_hash** και το **password_verify**, ώστε οι κωδικοί να αποθηκεύονται με ασφάλεια.

Τέλος, μέσω των API endpoints, δόθηκε η δυνατότητα στην εφαρμογή να χειρίζεται με ξεκάθαρο τρόπο τις διάφορες ενέργειες χρηστών που καλούνται από το frontend κομμάτι. Με τον συνδυασμό όλων των παραπάνω εργαλείων και τεχνολογιών, το backend κομμάτι εξασφάλισε την λειτουργικότητα και την αξιοπιστία της εφαρμογής, με επίκεντρο πάντα την ασφάλεια των δεδομένων και την αυξημένη αποδοτικότητα.

```
// Έλεγχος αν το request είναι POST
if ($_SERVER["REQUEST_METHOD"] == "POST") {

    $email = trim($_POST['email']);
    $password = trim($_POST['password']);

    // Έλεγχος στοιχείων αυθεντικοποίησης
    if (empty($email) || empty($password)) {
        echo json_encode(['success' => false, 'message' => 'Email and password are required!']);
        exit;
    }

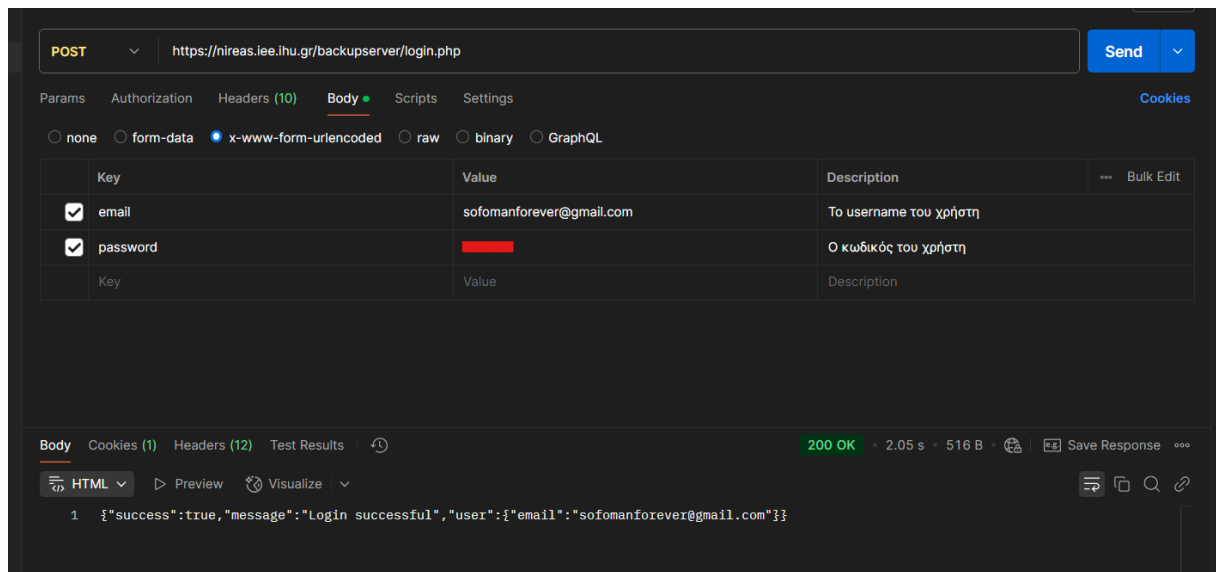
    try {
        $stmt = $pdo->prepare("SELECT * FROM users WHERE email = ?");
        $stmt->execute([$email]);
        $user = $stmt->fetch(PDO::FETCH_ASSOC);

        // Έλεγχος κωδικού
        if ($user && password_verify($password, $user['passwd'])) {
            // Αποθήκευση στοιχείων χρήστη στο session
            $_SESSION['user_id'] = $user['email'];
            $_SESSION['email'] = $user['email'];

            echo json_encode(['success' => true, 'message' => 'Login successful', 'user' => ['email' => $user['email']]]);
        } else {
            echo json_encode(['success' => false, 'message' => 'Invalid Email or Password']);
        }
    } catch (PDOException $e) {
        echo json_encode(['success' => false, 'message' => 'Database error: ' . $e->getMessage()]);
    }
    exit;
}
?>
```

Εικόνα 4.4.1: Διαχείριση αιτήματος POST

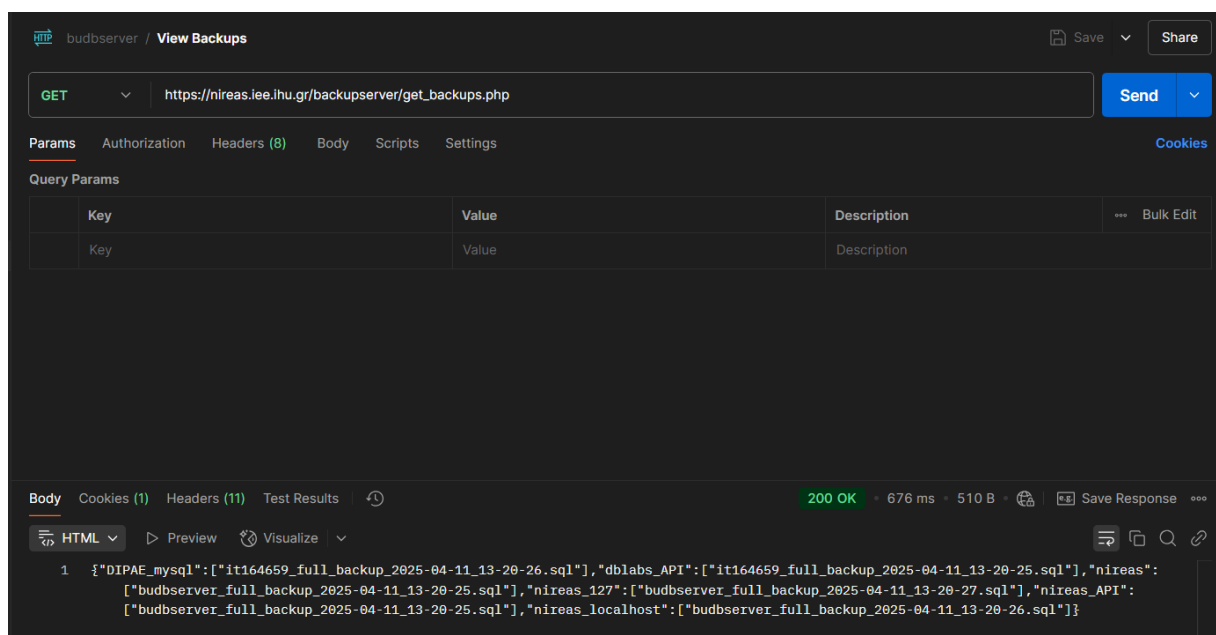
Όπως φαίνεται στην εικόνα 4.4.1, πρώτα γίνεται έλεγχος αν έχουν σταλεί τα απαραίτητα πεδία (στην περίπτωση αυτή email και κωδικός πρόσβασης). Στην συνέχεια γίνεται χρήση προετοιμασμένων εντολών (prepared statements) για την ανάκτηση των στοιχείων του χρήστη από την βάση, αποφεύγοντας έτσι sql injections. Με την συνάρτηση `password_verify` συγκρίνεται ο καταχωρημένος κώδικας με τον αποθηκευμένο, ενώ σε περίπτωση επιτυχούς ταυτοποίησης ξεκινάει το session με τα αντίστοιχα δεδομένα του χρήστη. Η παραπάνω επικοινωνία γίνεται μέσω JSON για άμεση και κατανοητή απόκριση στον client.



Εικόνα 4.4.2: Postman POST request

Για την δοκιμή και επαλήθευση των API endpoints, χρησιμοποίησα το Postman[53], το οποίο επιτρέπει την αποστολή POST αιτημάτων με μεταβλητά δεδομένα, την παρακολούθηση των απαντήσεων σε JSON μορφή για την πιο άμεση εντόπιση σφαλμάτων. Στην εικόνα 4.4.2 εκτέλεσα ένα αίτημα POST για επαλήθευση στοιχείων εισόδου στην εφαρμογή και δίνοντας τα σωστά στοιχεία έλαβα μήνυμα επιτυχίας OK με το αντίστοιχο μήνυμα “login successful”.

Μετά το επιτυχημένο login μέσω του API endpoint παραπάνω, εκτελώ ένα GET αίτημα, για να εμφανίσω τα αποθηκευμένα μου αντίγραφα ασφαλείας.



Εικόνα 4.4.3: Postman GET request

Αντίστοιχα και εδώ η απάντηση είναι σε JSON δεδομένα και εμφανίζονται όλα τα αποθηκευμένα αντίγραφα ασφαλείας. Ο κώδικας που χρησιμοποιήθηκε για να εμφανίζονται τα δεδομένα με αυτή την μορφή αναγράφεται στην παρακάτω εικόνα 4.4.4

```
κ?php
include 'db_connection.php';
session_start();

if (!isset($_SESSION['email'])) {
    echo json_encode(["success" => false, "message" => "Unauthorized access"]);
    exit;
}

$email = $_SESSION['email'];
$sanitizedEmail = str_replace(['@', '.'], '_', $email);
$userBackupDir = "backups/{$sanitizedEmail}/";

function getBackupsByConnection($baseDir) {
    $backups = [];
    if (is_dir($baseDir)) {
        $connections = array_filter(scandir($baseDir), function ($item) use ($baseDir) {
            return $item !== '.' && $item !== '..' && is_dir($baseDir . $item);
        });
        foreach ($connections as $connection) {
            $connectionDir = $baseDir . $connection . '/';
            $files = array_filter(scandir($connectionDir), function ($file) use ($connectionDir) {
                return is_file($connectionDir . $file);
            });
            $backups[$connection] = array_values($files);
        }
    }
    return $backups;
}

$backups = getBackupsByConnection($userBackupDir);
echo json_encode($backups);
?>
```

Εικόνα 4.4.4: Κώδικας υλοποίησης απάντηση αιτήματος GET

4.5.1 Δημιουργία αντιγράφου ασφαλείας

Επόμενο σημαντικό κομμάτι του backend κομματιού της εφαρμογής είναι η εκτέλεση της εντολής δημιουργίας αντιγράφων ασφαλείας τόσο σε MySQL βάσεις όσο και σε PostgreSQL. Είναι ουσιαστικά το σημαντικότερο κομμάτι της εφαρμογής, καθώς αποτελεί την ‘καρδιά’ της λειτουργίας αυτής.

Αρχικά, ορίζουμε μια συνάρτηση την οποία θα χρησιμοποιούμε για την αποκρυπτογράφηση των κωδικών πρόσβασης που έχουν αποθηκευτεί με ασφάλεια, με σκοπό να χρησιμοποιηθούν μόνο κατά την διάρκεια της εκτέλεσης.

```
define('ENCRYPTION_KEY', 'your-secure-encryption-key');

function decryptPassword($encryptedPassword) {
    return openssl_decrypt($encryptedPassword, 'AES-128-ECB', ENCRYPTION_KEY);
}
```

Εικόνα 4.4.5: Συνάρτηση αποκρυπτογράφησης κωδικών

Έπειτα μόλις αποσταλεί ένα POST αίτημα, γίνεται λήψη των στοιχείων της εκάστοτε σύνδεσης και αναζητούνται τα αντίστοιχα δεδομένα στην βάση. Με αυτό το statement γίνεται η αναζήτηση:

```
$stmt = $pdo->prepare("SELECT host, db_username, db_password,
db_name, connection_name, email, ssh_host, ssh_username, ssh_password,
ssh_port, db_type FROM user_db_connections WHERE connection_id = ?");
```

SQL Statement για αναζήτηση στοιχείων σύνδεσης

Εφόσον εντοπιστούν, αποκρυπτογραφούνται τα στοιχεία αυθεντικοποίησης της βάσης δεδομένων αλλά και του SSH[48] (αν ο χρήστης τα έχει δώσει κατά την δημιουργία της σύνδεσης του), καθώς σε αρκετές περιπτώσεις το backup εκτελείται μέσω απομακρυσμένου server ως εξής:

```

$stmt->execute([$connectionId]);
$connectionDetails = $stmt->fetch(PDO::FETCH_ASSOC);

if ($connectionDetails) {
    $host = $connectionDetails['host'];
    $username = $connectionDetails['db_username'];
    $encryptedPassword = $connectionDetails['db_password'];
    $database = $connectionDetails['db_name'];
    $connectionName = $connectionDetails['connection_name'];
    $email = $connectionDetails['email'];
    $password = decryptPassword($encryptedPassword); // αποκρυπτογράφηση κωδικού βάσης

    $encryptedSshPassword = $connectionDetails['ssh_password'];
    $sshPassword = decryptPassword($encryptedSshPassword); // αποκρυπτογράφηση SSH κωδικού

    $sshHost = $connectionDetails['ssh_host'];
    $sshUsername = $connectionDetails['ssh_username'];
    $sshPort = $connectionDetails['ssh_port'] ?: 22;
    $dbType = $connectionDetails['db_type'];

```

Εικόνα 4.4.6: Στοιχεία σύνδεσης με αποκρυπτογράφηση κωδικών πρόσβασης

Στην συνέχεια δημιουργείται ένας φάκελος αποθήκευσης με ονοματολογία αντίστοιχη του email του χρήστη, προκειμένου να ταξινομούνται αποτελεσματικότερα τα αντίγραφα ασφαλείας.

```

// δημιουργία φακέλου χρήστη με όνομα από το email του
$userFolder = 'backups/' . preg_replace('/^[a-zA-Z0-9]/', '_', $email);
if (!is_dir($userFolder)) {
    mkdir($userFolder, 0777, true);
}

// δημιουργία φακέλου σύνδεσης μέσα στον φάκελο του χρήστη
$connectionFolder = $userFolder . '/' . preg_replace('/^[a-zA-Z0-9]/', '_', $connectionName);
if (!is_dir($connectionFolder)) {
    mkdir($connectionFolder, 0777, true);
}

```

Εικόνα 4.4.7: Δημιουργία φακέλου σύνδεσης

Ταυτόχρονα, διαμορφώνουμε και την ονοματολογία του αντιγράφου ασφαλείας που πρόκειται να δημιουργηθεί, εξασφαλίζοντας ότι το όνομα θα αποτελείται από το όνομα της βάσης, το είδος του backup και την χρονική στιγμή (μέρα και ώρα) που θα δημιουργηθεί. Καθορίζουμε και το path του φακέλου της αντίστοιχης σύνδεσης για την ταξινομημένη αποθήκευση του.

```

// δημιουργία ονοματος αντιγράφου ασφαλείας μέσα στον φάκελο
$timestamp = date("Y-m-d_H-i-s");
$backupFileName = $database . "_" . $backupType . "_backup_" . $timestamp . ".sql";
$backupFilePath = $connectionFolder . "/" . $backupFileName;

```

Εικόνα 4.4.8: Δημιουργία ονόματος αντιγράφου ασφαλείας

Ανάλογα με τον τύπο της βάσης (\$dbType) και την ύπαρξη ή μη ssh σύνδεσης, διαμορφώνεται η κατάλληλη εντολή mysqldump ή pg_dump, αφού προηγηθεί η χρήση της εντολής sshpass για απομακρυσμένη πρόσβαση.

```
$command = '';

if ($dbType === 'mysql') {

    if ($sshHost && $sshUsername && $sshPassword &&
$backupType === 'full') {

        // sshpass για MySQL

        $command = sprintf(

            "sshpass -p '%s' ssh -o
StrictHostKeyChecking=no -p 22 -v %s@%s 'mysqldump --no-tablespaces -u
%s -p%s -h %s %s' > %s",

            escapeshellarg($sshPassword),

            escapeshellarg($sshUsername),

            escapeshellarg($sshHost),

            escapeshellarg($username),

            escapeshellarg($password),

            escapeshellarg($host),

            escapeshellarg($database),

            escapeshellarg($backupFilePath)

        );

    } elseif ($backupType === 'full') {

        $command = "mysqldump -u " .
escapeshellarg($username) . " -p" . escapeshellarg($password) . " " .
escapeshellarg($database) . " > " . escapeshellarg($backupFilePath);

    }

}
```

Σύνταξη εντολής δημιουργίας backup MySQL βάσης

```

} elseif ($dbType === 'postgres') {
    if ($sshHost && $sshUsername && $sshPassword &&
$backupType === 'full') {
        // sshpass για PostgreSQL

        $command = sprintf(
            "sshpass -p '%s' ssh -o
StrictHostKeyChecking=no -p 22 %s@%s 'export PGPASSWORD=%s; pg_dump -U
%s -h %s -d %s' > %s",
            escapeshellarg($sshPassword),
            escapeshellarg($sshUsername),
            escapeshellarg($sshHost),
            escapeshellarg($password),
            escapeshellarg($username),
            escapeshellarg($host),
            escapeshellarg($database),
            escapeshellarg($backupFilePath)
        );
    } elseif ($backupType === 'full') {
        $command = "pg_dump -U " .
escapeshellarg($username) . " -h " . escapeshellarg($host) . " " .
escapeshellarg($database) . " > " . escapeshellarg($backupFilePath);
    }
}

```

Σύνταξη εντολής δημιουργίας backup MySQL βάσης

Η τελική εντολή εκτελείται μέσω “exec()” .

```
exec($command . " 2>&1", $output, $return_var);
```

Σύνταξη εντολής εκτέλεσης exec();

Το αποτέλεσμα αξιολογείται και αν η εκτέλεση της εντολής ήταν επιτυχής, τότε το backup καταγράφεται στην βάση ως ιστορικό μέσω SQL INSERT statement, μαζί με τον τύπο του backup αλλά και την ώρα που δημιουργήθηκε. Σε περίπτωση αποτυχίας, εμφανίζεται αντίστοιχο μήνυμα λάθους για ενημέρωση του χρήστη.

```
if ($return_var === 0) {

    $stmt = $pdo->prepare("INSERT INTO backup_logs
(file_name, email, backup_type, last_backup) VALUES (:file_name,
:email, :backup_type, NOW())");

    $stmt->execute([

        ':file_name' => $backupFileName,

        ':email' => $email,

        ':backup_type' => $backupType

    ]);

    echo json_encode(['success' => true, 'message' =>
"Backup successful. Backup saved to: " .
htmlspecialchars($backupFilePath)]);

} else {

    echo json_encode(['success' => false, 'message' =>
"Error executing backup command: " . implode("\n", $output)]);

}

} else {

    echo json_encode(['success' => false, 'message' =>
"Connection details not found for the given ID."]);

}

} catch (PDOException $e) {

    echo json_encode(['success' => false, 'message' => "Database
error: " . $e->getMessage()]);

}

}
```

Σύνταξη εντολής εκτέλεσης exec();

Αν η μεταβλητή **\$return_var** ισούται με 0 , αυτό σημαίνει ότι η εντολή εκτελέστηκε επιτυχώς χωρίς σφάλμα. Σε αυτή την περίπτωση εισάγεται μια νέα εγγραφή στον πίνακα **backup_logs**, που περιλαμβάνει το όνομα του αρχείου backup, το email του χρήστη που το δημιούργησε, τον τύπο του αλλά και την ημερομηνία και ώρα δημιουργίας του. Έπειτα επιστρέφεται σε μορφή JSON μήνυμα επιτυχίας.

Στην περίπτωση που η τιμή της μεταβλητής **\$return_var** είναι διάφορη του μηδενός, τότε εμφανίζεται μήνυμα αποτυχίας, μαζί με περιγραφή του σφάλματος από την έξοδο του συστήματος.

Τέλος, αν δεν βρεθούν στοιχεία σύνδεσης βάσης δεδομένων με το δοθέν **connection_id**, ο χρήστης ενημερώνεται αντίστοιχα. Περιλαμβάνονται και try_catch statements για την καλύτερη διαχείριση πιθανών σφαλμάτων κατά την εκτέλεση των SQL ερωτημάτων.

Με τον συνδυασμό όλων των παραπάνω βημάτων, δημιουργούνται αντίγραφα ασφαλείας και για τα 2 είδη βάσεων δεδομένων, ενώ ταυτόχρονα αρχειοθετούνται με τέτοιο τρόπο ώστε να είναι πιο ταξινομημένη η αποθήκευση τους και να μπορεί ο χρήστης να βρίσκει ευκολότερα το αντίγραφο που τον ενδιαφέρει ονοματικά και χρονικά.

4.6 Github Repository

Όλος ο πηγαίος κώδικας της εφαρμογής μου έχει αναρτηθεί σε δημόσιο αποθετήριο στο github.

Μέσα σε αυτό περιέχονται όλα τα αρχεία της εφαρμογής, χωρισμένα σε φακέλους ανάλογα με την λειτουργία τους (frontend, backend, logs, docs), για να διευκολύνεται η κατανόηση της δομής του προγράμματος.

Η ύπαρξη του αποθετηρίου επιτρέπει την επαναχρησιμοποίηση ή επέκταση της εφαρμογής από άλλους χρήστες στο μέλλον , ενώ παράλληλα λειτουργούσε ως τεκμήριο της προόδου και της μεθοδολογίας και του συνόλου των διαφορετικών τεχνολογιών που χρησιμοποίησα.

Το αποθετήριο της εφαρμογής μου είναι διαθέσιμο στον παρακάτω σύνδεσμο:

<https://github.com/sofo-roth/BUDBServer-Thesis-Project.git>

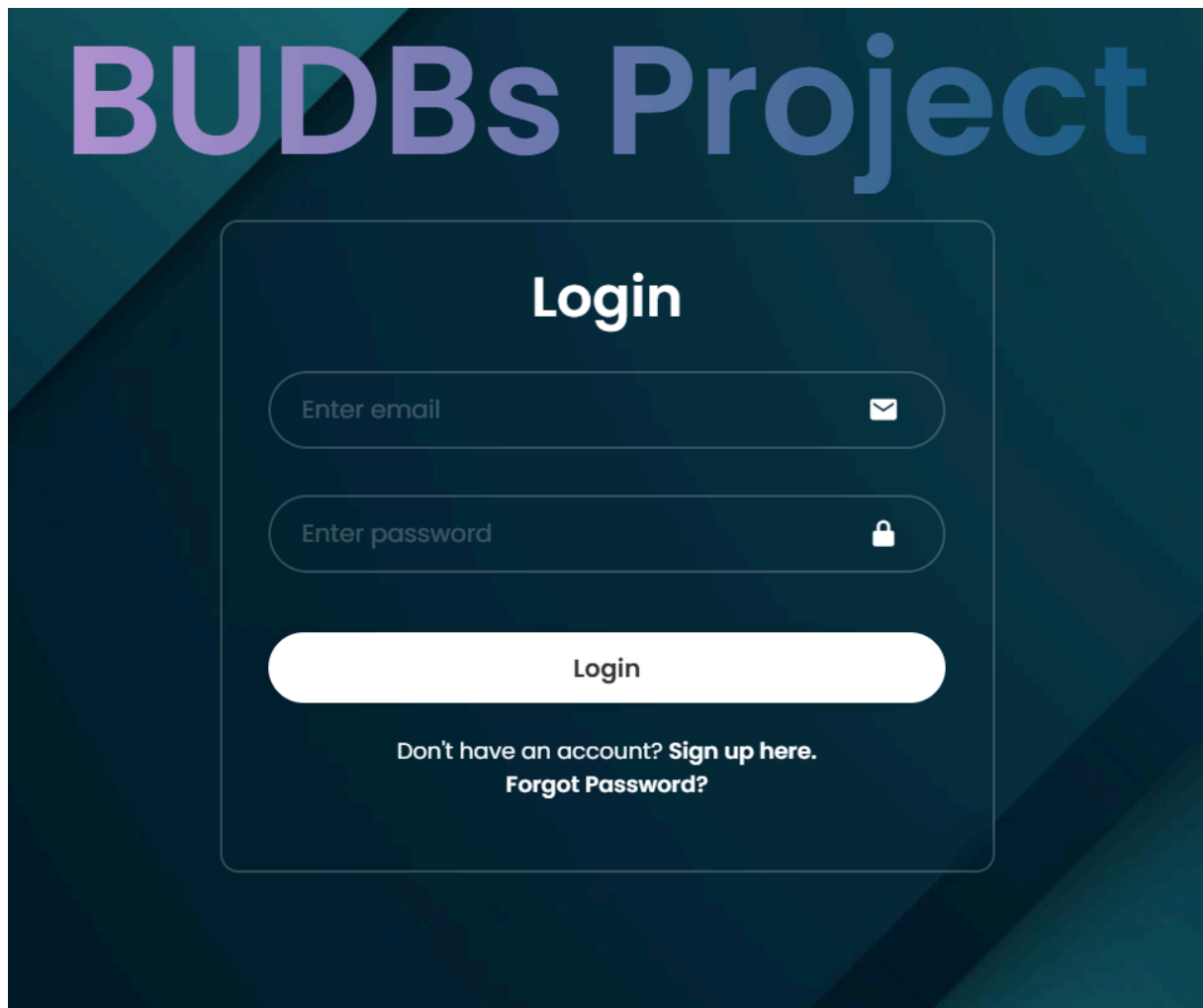
Κεφάλαιο 5ο: Παρουσίαση BUDBServer

5.1 Εισαγωγή - Περιγραφή

Η BUDBServer web εφαρμογή αυτή σχεδιάστηκε με στόχο την απλοποίηση της διαδικασίας δημιουργίας αντιγράφων ασφαλείας για βάσεις δεδομένων MySQL[36][39] και PostgreSQL[38]. Παρέχει στους χρήστες τη δυνατότητα να δημιουργούν και να αποθηκεύουν τις συνδέσεις τους με τις βάσεις δεδομένων, χρησιμοποιώντας αυθεντικοποιημένα στοιχεία τα οποία αποθηκεύονται στο προσωπικό τους προφίλ αποκρυπτογραφημένα και με ασφάλεια. Η εφαρμογή προσφέρει ένα φιλικό περιβάλλον για την παρακολούθηση, λήψη και διαγραφή των backups. Απευθύνεται κυρίως σε διαχειριστές και προγραμματιστές που επιθυμούν αξιόπιστη και εύχρηστη λύση για τη δημιουργία και διαχείριση αντιγράφων ασφαλείας.

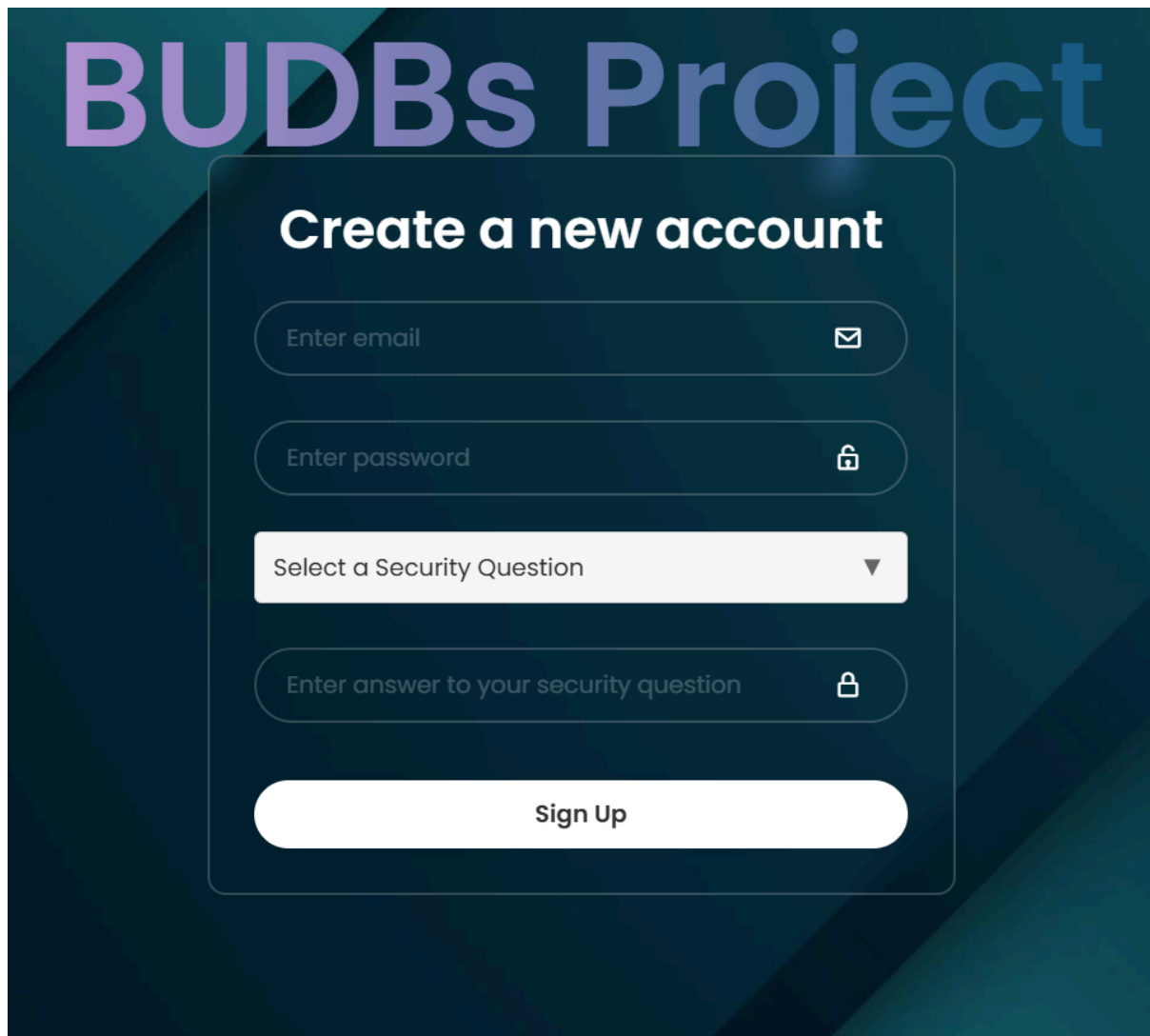
5.2 Λειτουργίες της εφαρμογής

Ξεκινώντας, ακολουθούμε τον σύνδεσμο <https://nireas.iee.ihu.gr/backupserver/login.php>, όπου μας εμφανίζει την φόρμα login της εφαρμογής μας:



Εικόνα 5.2.1: Φόρμα Login

Αν είμαστε καινούργιοι χρήστες και δεν έχουμε κάποιον λογαριασμό, μπορούμε να ακολουθήσουμε τον σύνδεσμο “**Sign up here**” προκειμένου να ανακατευθυνθούμε στην φόρμα δημιουργίας λογαριασμού (Sign up form) :



The image shows a sign-up form for the 'BUDBs Project'. The form is titled 'Create a new account' and is set against a dark blue background with the project name 'BUDBs Project' in large, light blue letters. The form itself is a white rounded rectangle containing the following elements: an email input field with an envelope icon, a password input field with a lock icon, a dropdown menu for 'Select a Security Question', another password input field for the answer, and a large white 'Sign Up' button at the bottom.

Εικόνα 5.2.2: Φόρμα Sign Up

Σε αυτήν την φόρμα μπορούμε να δημιουργήσουμε τον καινούργιο μας λογαριασμό, με όνομα χρήστη το email μας, κωδικό πρόσβασης και επιλογή ερώτησης ασφαλείας, ώστε στην φόρμα επαναφοράς κωδικού πρόσβασης, που θα αναλύσουμε παρακάτω, να μπορούμε να την χρησιμοποιήσουμε προκειμένου να εξακριβώνεται επιτυχώς ο χρήστης που ξέχασε τον κωδικό του.

Όσον αφορά την ερώτηση ασφαλείας, οι επιλογές που έχει στην διάθεση του ο χρήστης είναι:

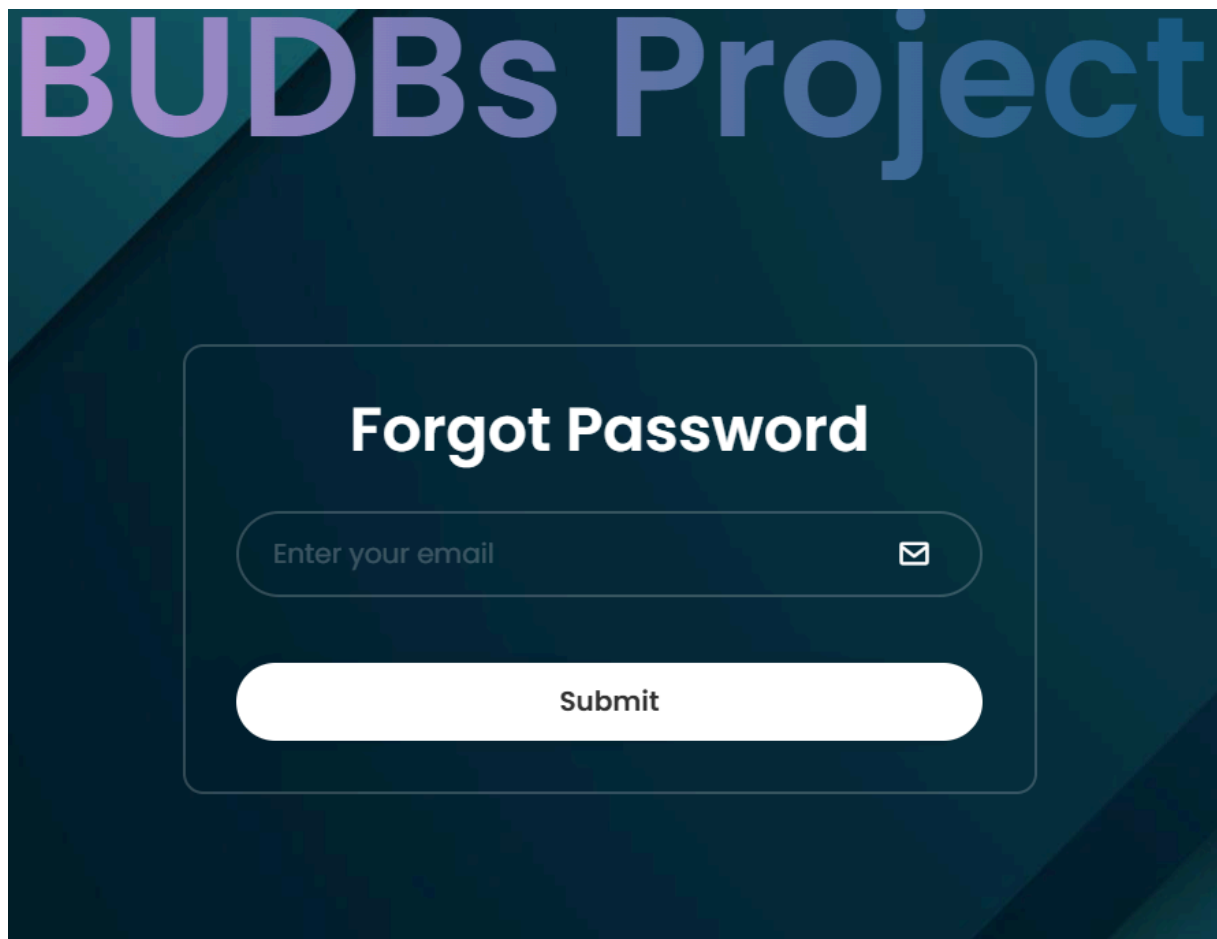
- Ποιό είναι το όνομα του πρώτου σου κατοικιδίου ;
- Ποιό είναι το πατρικό όνομα της μητέρας σου ;
- Ποιό είναι το όνομα της πόλης που κατάγεται ;

Η απάντηση του χρήστη μπορεί να είναι οποιαδήποτε αλφαριθμητική τιμή επιθυμεί , η τιμή της οποίας μαζί με τον κωδικό πρόσβασης αποθηκεύονται κρυπτογραφημένα. Εάν ο χρήστης δεν συμπληρώσει όλα τα πεδία της φόρμας και πατήσει το sign up κουμπί τότε εμφανίζεται σχετικό

μήνυμα που του υποδεικνύει τα κενά πεδία και την υποχρεωτικότητα συμπλήρωσης όλων, προκειμένου να προχωρήσει σε δημιουργία λογαριασμού. Επίσης, εφόσον η δομή της εφαρμογής έχει διαμορφωθεί με τέτοιο τρόπο ώστε να δουλεύει με emails ως usernames, αν ο χρήστης δεν έχει δώσει email στο username πεδίο, τότε βγαίνει αντίστοιχο μήνυμα που τον ενημερώνει ότι δεν μπορεί να χρησιμοποιήσει διαφορετική μορφή username πέρα από email.

Εφόσον συμπληρωθούν όλα τα πεδία της φόρμας με την σωστή μορφή, ο χρήστης κάνει sign up και ανακατευθύνεται στην φόρμα login, προκειμένου να συνδεθεί κανονικά στο προφίλ του που μόλις έχει δημιουργήσει και να ξεκινήσει να διαμορφώνει τις συνδέσεις του.

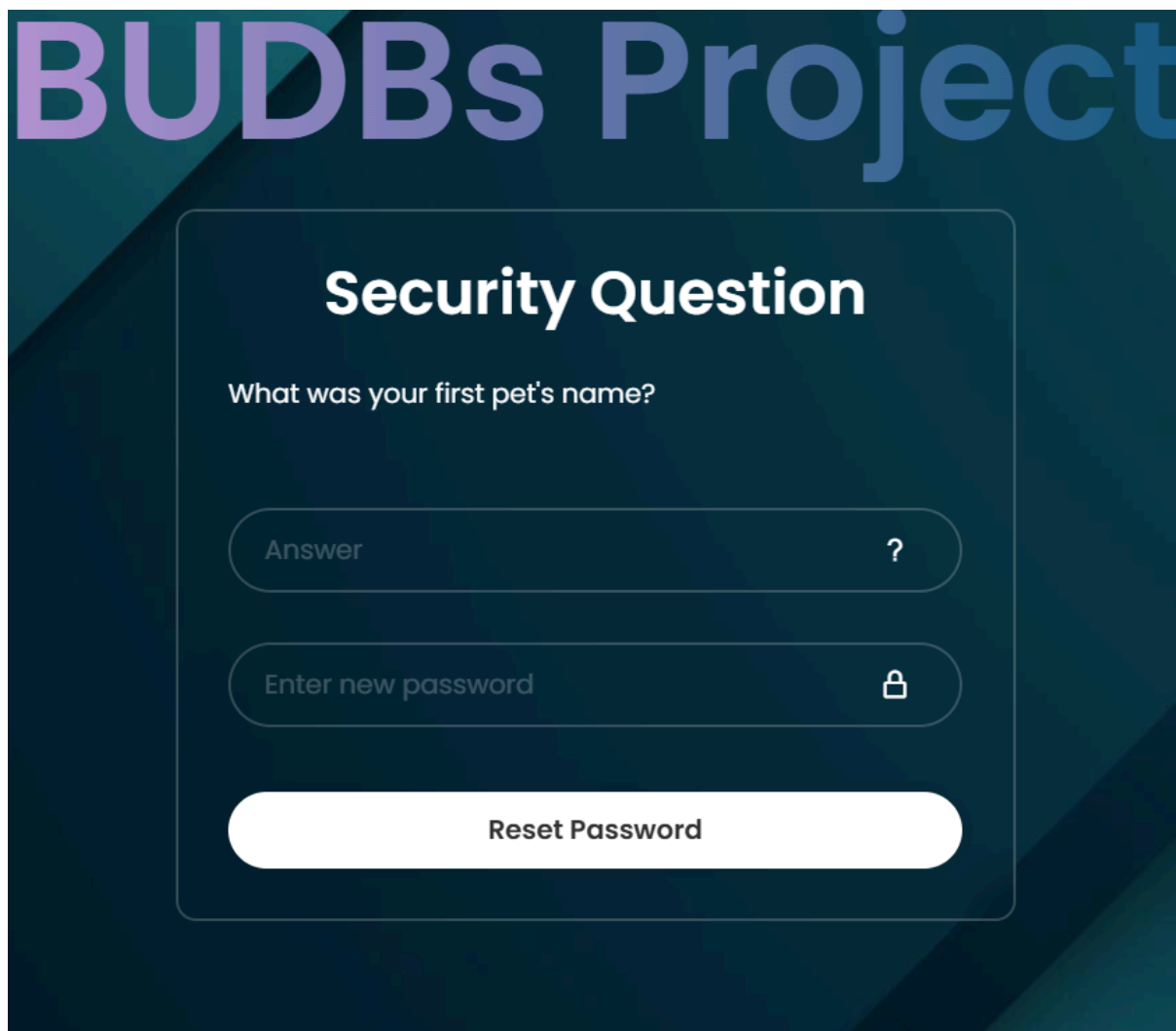
Στην περίπτωση που ο χρήστης έχει ήδη λογαριασμό αλλά για οποιοδήποτε λόγο έχει ξεχάσει τον κωδικό πρόσβασης του, μπορεί να ακολουθήσει τον σύνδεσμο [“Forgot Password?”](#) της φόρμας login όπως αναγράφεται στην εικόνα 5.2.1 . Με τον σύνδεσμο αυτό, ανακατευθύνεται στην φόρμα επαναφοράς κωδικού πρόσβασης, όπου προκειμένου να προχωρήσει στην δημιουργία καινούργιου κωδικού πρόσβασης, θα πρέπει να απαντήσει σωστά στην ερώτηση ασφαλείας που είχε προκαθορίσει κατά την διαδικασία δημιουργίας λογαριασμού.

The image shows a web interface for the 'BUDBs Project'. At the top, the title 'BUDBs Project' is displayed in large, stylized letters. Below this, there is a 'Forgot Password' form. The form has a title 'Forgot Password' in bold. Below the title is a text input field with the placeholder text 'Enter your email' and an email icon. Below the input field is a 'Submit' button.

Εικόνα 5.2.3: Φόρμα επαναφοράς κωδικού πρόσβασης, προσθήκη email

Προκειμένου να προχωρήσει στην απάντηση της ερώτησης ασφαλείας, θα πρέπει να εισάγει το email που χρησιμοποιεί ως username στην εφαρμογή, ώστε να εξακριβωθεί αν ο χρήστης αυτός όντως υπάρχει. Στην περίπτωση που δεν υπάρχει, εμφανίζεται αντίστοιχο μήνυμα.

Μόλις εξακριβωθεί το email, εμφανίζεται η φόρμα υποβολής απάντησης στην ερώτηση ασφαλείας όπως φαίνεται παρακάτω:



BUDBs Project

Security Question

What was your first pet's name?

Answer ?

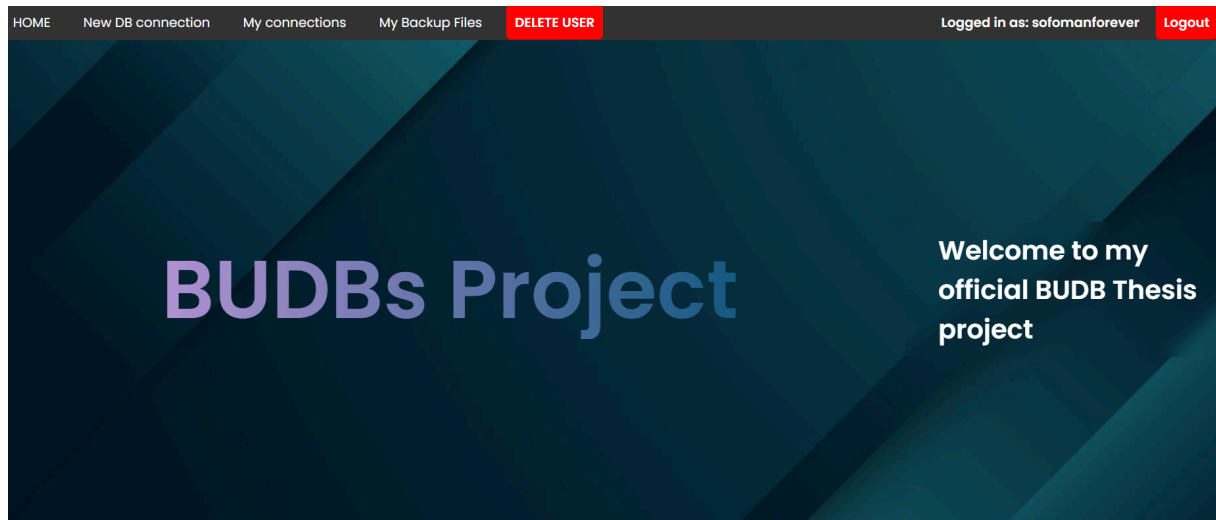
Enter new password

Reset Password

Εικόνα 5.2.4: Φόρμα υποβολής απάντησης ασφαλείας και δημιουργίας κωδικού πρόσβασης

Στην παραπάνω φόρμα, ο χρήστης καλείται να εισάγει την απάντηση που είχε αποθηκεύσει στην ερώτηση ασφαλείας που είχε επιλέξει, καθώς και να εισάγει τον καινούργιο του κωδικό πρόσβασης. Στην περίπτωση που η απάντηση ασφαλείας είναι λανθασμένη, εμφανίζεται σχετική ειδοποίηση, ενημερώνοντας τον ότι δόθηκε λανθασμένη απάντηση και σαφώς δεν μπορεί να συνεχιστεί η επαναφορά κωδικού μέχρι να εισάγει την σωστή. Με το που εισάγει την σωστή απάντηση και πληκτρολογήσει τον νέο κωδικό πρόσβασης του, βγαίνει ειδοποίηση επιτυχίας επαναφοράς κωδικού πρόσβασης και ανακατευθύνεται στην φόρμα login για να προχωρήσει με την είσοδο του στην εφαρμογή.

Με την σωστή αυθεντικοποίηση του χρήστη στην φόρμα login, ανακατευθύνεται στο dashboard, όπου από κει χρησιμοποιώντας τις διαθέσιμες επιλογές από το navigation bar στην κορυφή της εφαρμογής, μπορεί να δημιουργήσει καινούργιες συνδέσεις, να εκτελέσει διαδικασίες δημιουργίας αντιγράφων ασφαλείας των βάσεων στις οποίες έχει πρόσβαση, να δει τα διαθέσιμα του αντίγραφα ασφαλείας, να τα κατεβάσει τοπικά ή να τα διαγράψει, καθώς και αν επιθυμεί να διαγράψει ολόκληρο το προφίλ του.



Εικόνα 5.2.5: Αρχική σελίδα χρήστη

Προχωράμε στην δημιουργία συνδέσεων με τις βάσεις δεδομένων στις οποίες ο εκάστοτε χρήστης έχει πρόσβαση. Με την επιλογή του συνδέσμου “New DB connection”, γίνεται ανακατεύθυνση στην φόρμα δημιουργίας συνδέσεων με τα εξής πεδία:

Add a New Database Connection

Database Type
☒ MySQL ☐ PostgreSQL

Connection Name

Host

DB Username

DB Password

Database Name

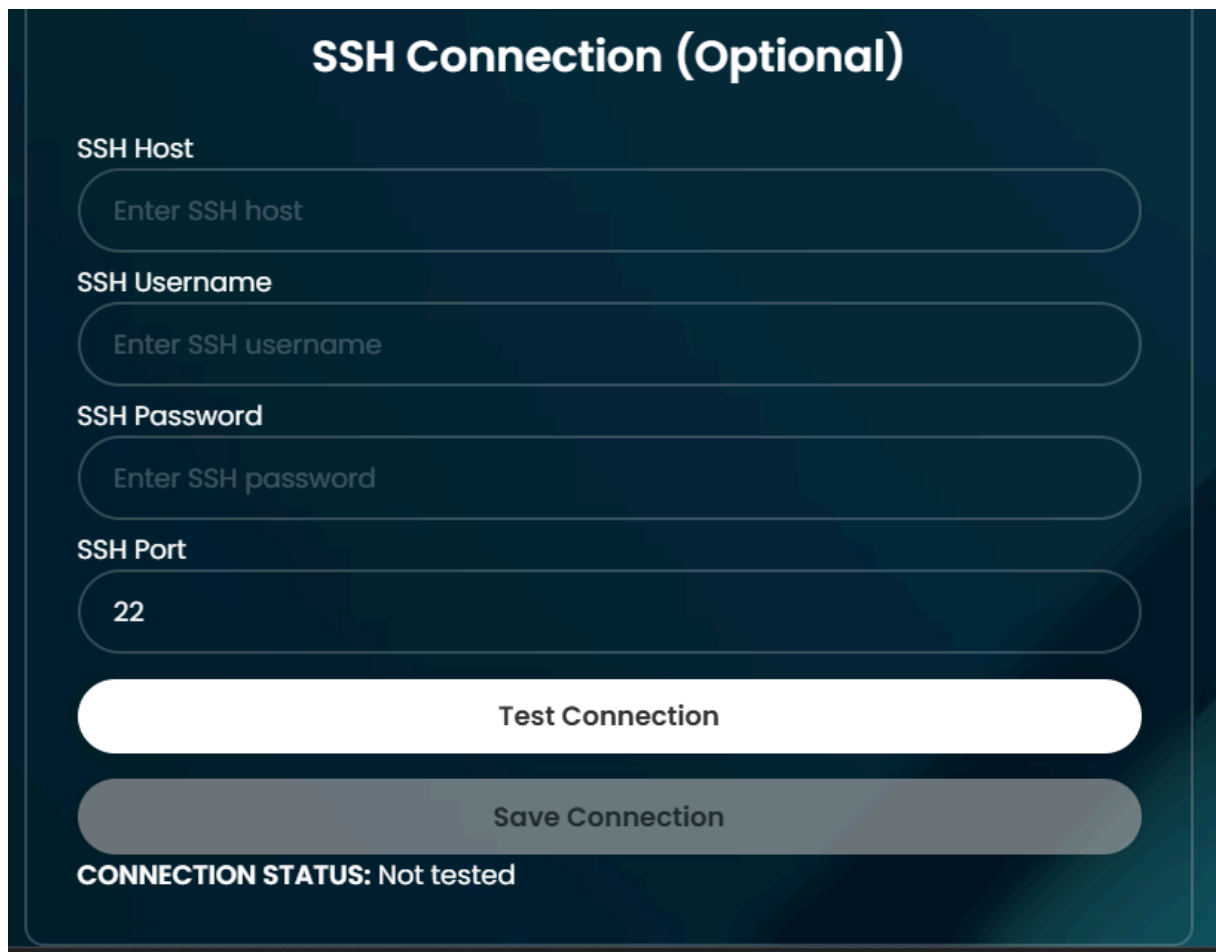
Εικόνα 5.2.6: Φόρμα δημιουργίας καινούργιας σύνδεσης

Ανάλογα το είδος της βάσης με την οποία ο χρήστης επιθυμεί να δημιουργήσει σύνδεση μπορεί να επιλέξει αν η βάση είναι MySQL ή PostgreSQL, έτσι ώστε οι διαδικασίες δημιουργίας αντιγράφων και οι εντολές που αποστέλλονται από την εφαρμογή στις βάσεις αυτές μέσω php να είναι οι αντίστοιχες σωστές.

Ο χρήστης μπορεί να χρησιμοποιήσει οποιοδήποτε όνομα επιθυμεί στο πεδίο Connection Name, δίνοντας του την επιλογή να τα διαμορφώσει με τέτοιο τρόπο ώστε να μπορεί να ξεχωρίζει τις συνδέσεις του, ειδικότερα αν οι βάσεις στις οποίες πραγματοποιεί συνδέσεις έχουν κοινό φορέα. Αυτό το όνομα επίσης χρησιμοποιείται για τον φάκελο που δημιουργείται αυτόματα μόλις ο χρήστης επιτυχώς δημιουργήσει μια σύνδεση, στον οποίο αποθηκεύονται τα αντίστοιχα αντίγραφα και εμφανίζονται κατηγοριοποιημένα στην φόρμα των αντιγράφων του εκάστοτε χρήστη.

Τα υπόλοιπα πεδία της παραπάνω φόρμας συμπληρώνονται ακριβώς όπως ο χρήστης πραγματοποιεί την σύνδεση του στην βάση εκτός εφαρμογής. Οποιοδήποτε λάθος κάνει σε αυτά τα πεδία, του εμφανίζεται το αντίστοιχο μήνυμα που θα του εμφανιζόταν στην κονσόλα, βοηθώντας τον έτσι να ανιχνεύσει πιο εύκολα το λάθος.

Ωστόσο, οι περισσότερες συνδέσεις προαπαιτούν να πραγματοποιηθεί μια SSH σύνδεση, προκειμένου να μπορέσει ο χρήστης να έχει πρόσβαση στην εκάστοτε του βάση. Μέσω των πεδίων της παρακάτω φόρμας, σε ένα ποιο user friendly περιβάλλον, ο χρήστης εισάγει τα στοιχεία πραγματοποίησης ssh σύνδεσης όπως αντίστοιχα θα τα χρησιμοποιούσε και στην κονσόλα.



SSH Connection (Optional)

SSH Host
Enter SSH host

SSH Username
Enter SSH username

SSH Password
Enter SSH password

SSH Port
22

Test Connection

Save Connection

CONNECTION STATUS: Not tested

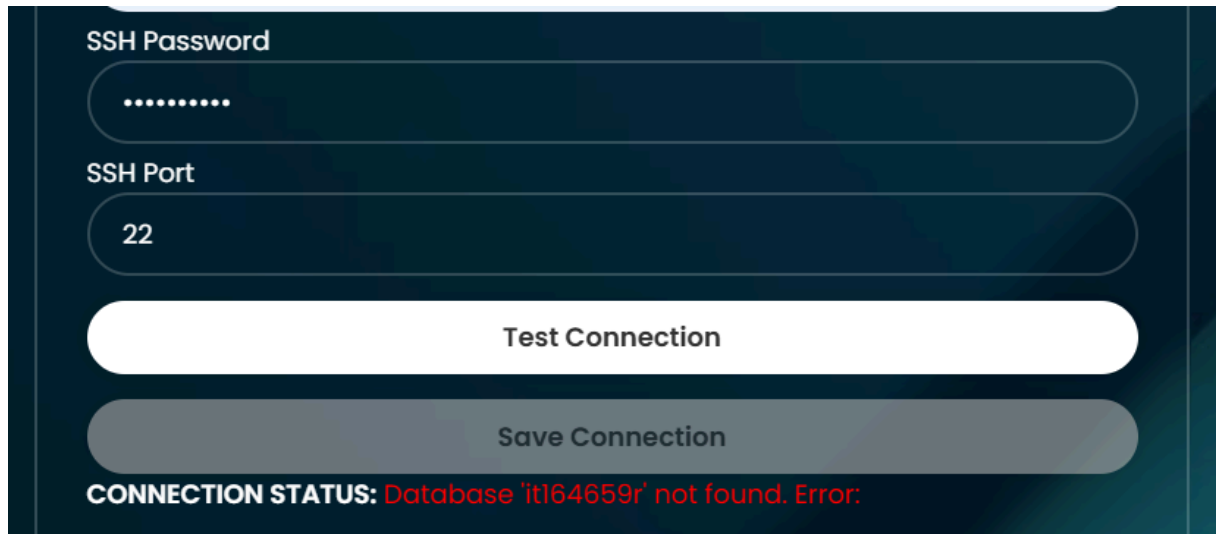
Εικόνα 5.2.7: Υποβολή στοιχείων για σύνδεση SSH

Με την πλειοψηφία των συνδέσεων ssh να χρησιμοποιούν το port 22, η τιμή του αντίστοιχου πεδίου της φόρμας αρχικοποιείται με την τιμή αυτή. Εννοείται πως ο χρήστης έχει την δυνατότητα, αν η σύνδεση που θέλει να πραγματοποιήσει χρησιμοποιεί διαφορετικό port, να αλλάξει την τιμή αυτή, διαμορφώνοντας την με τέτοιο τρόπο ώστε η ssh σύνδεση να πραγματοποιηθεί με επιτυχία.

Στο κάτω μέρος της φόρμας παρατηρούμε ότι υπάρχουν 2 κουμπιά και ένα connection status board.

Το κουμπί της αποθήκευσης της σύνδεσης δεν είναι διαθέσιμο, παρά μόνο εάν ο χρήστης με τα στοιχεία που έχει δώσει τόσο στα υποχρεωτικά όσο και στα προαιρετικά πεδία της φόρμας δημιουργίας σύνδεσης, καταφέρει πρώτα να πραγματοποιήσει μια δοκιμαστική σύνδεση. Εάν δινόταν η δυνατότητα στον χρήστη να αποθηκεύει τις καινούργιες του συνδέσεις χωρίς την πραγματοποίηση ελέγχου στα στοιχεία αυθεντικοποίησης που έχει δώσει, τότε θα ήταν αδύνατη η εκτέλεση των διαδικασιών δημιουργίας αντιγράφων, σε περίπτωση υπαρξης λάθους στα στοιχεία αυθεντικοποίησης.

ΠΑΡΑΔΕΙΓΜΑ ΛΑΘΟΥΣ 1



SSH Password

.....

SSH Port

22

Test Connection

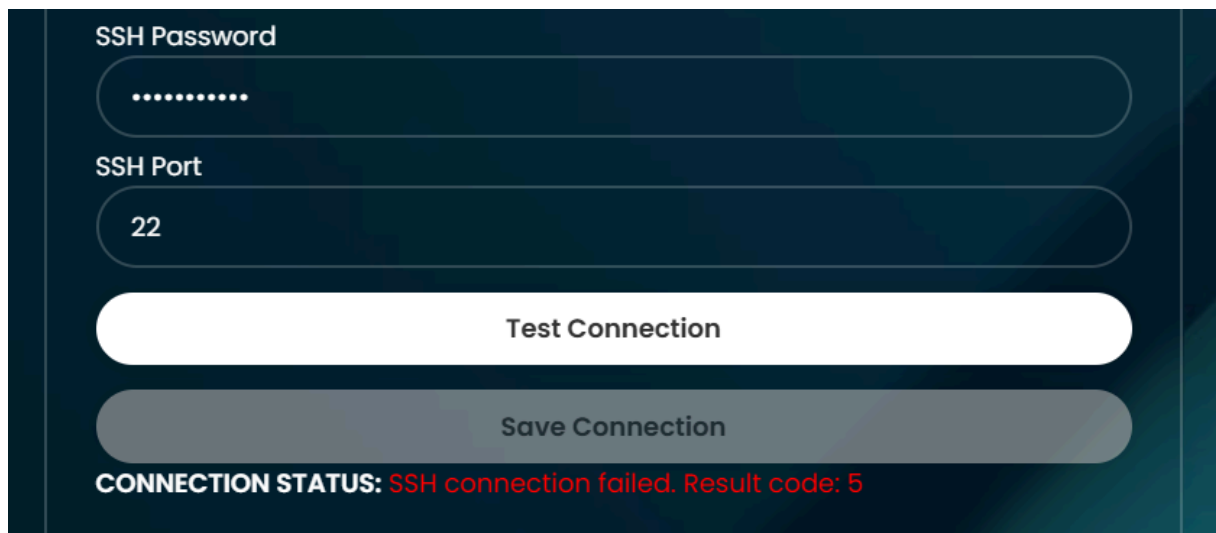
Save Connection

CONNECTION STATUS: Database 'ltl64659r' not found. Error:

Εικόνα 5.2.8: Σενάριο λάθους ονόματος βάσης

Στο παράδειγμα 1, έχω δώσει επίτηδες ένα όνομα βάσης που δεν υπάρχει. Με το τεστάρισμα της σύνδεσης, η κονσόλα μου εμφανίζει το σημείο στο οποίο βρέθηκε το λάθος, έτσι ώστε να μην χρειαστεί να αλλάξω και να συμπληρώσω από την αρχή όλα τα στοιχεία αλλά να επικεντρωθώ σε αυτό που προκαλεί το error.

ΠΑΡΑΔΕΙΓΜΑ ΛΑΘΟΥΣ 2



SSH Password

.....

SSH Port

22

Test Connection

Save Connection

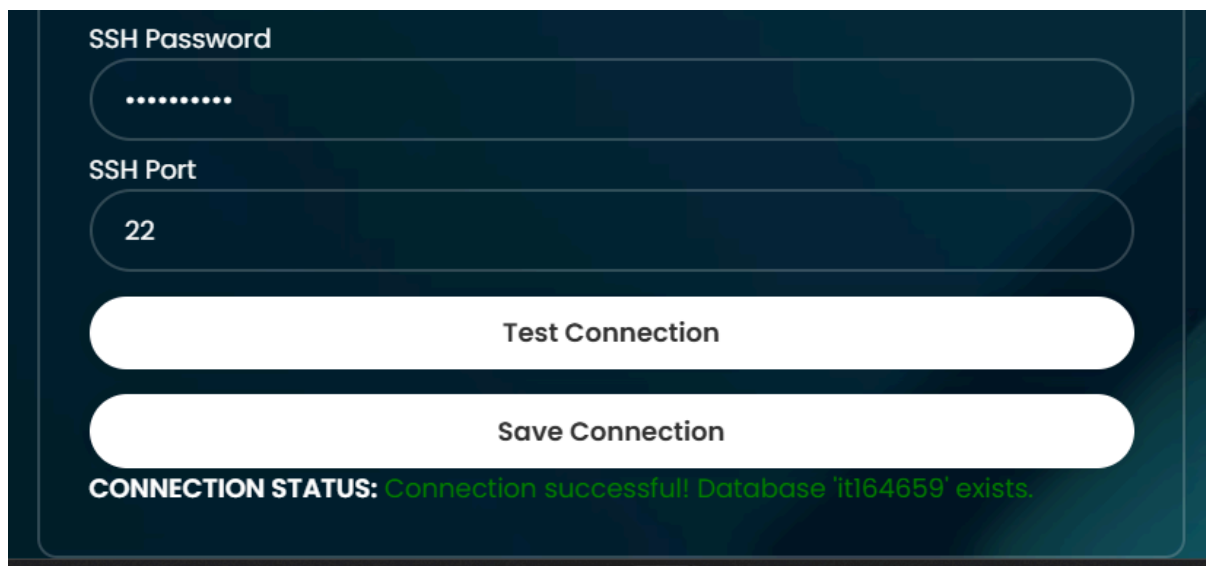
CONNECTION STATUS: SSH connection failed. Result code: 5

Εικόνα 5.2.9: Σενάριο λάθους κωδικού σύνδεσης SSH

Στο παράδειγμα 2, έχω δώσει επίτηδες λάθος κωδικό για την πραγματοποίηση της ssh σύνδεσης. Με το τεστάρισμα της σύνδεσης, η κονσόλα μου εμφανίζει τόσο το σημείο στο οποίο ανιχνεύθηκε το

λάθος, όσο και την τιμή του result code, που στην προκειμένη περίπτωση είναι 5, υποδεικνύοντας έτσι ότι το λάθος βρίσκεται στο πεδίο του κωδικού, καθώς του απαγορεύεται η πρόσβαση.

ΠΑΡΑΔΕΙΓΜΑ ΣΩΣΤΗΣ ΣΥΝΔΕΣΗΣ

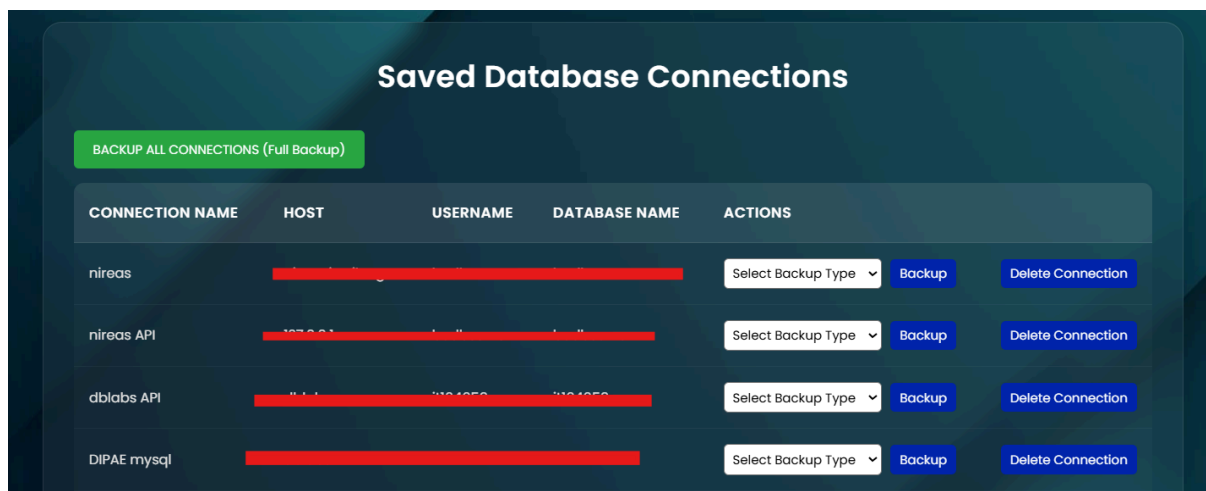


The image shows a dark-themed user interface for an SSH connection. It features two input fields: 'SSH Password' with masked characters (dots) and 'SSH Port' with the value '22'. Below these are two large, rounded buttons: 'Test Connection' and 'Save Connection'. At the bottom, a green status message reads: 'CONNECTION STATUS: Connection successful! Database 'it164659' exists.'

Εικόνα 5.2.10: Σενάριο υποβολής σωστών στοιχείων αυθεντικοποίησης

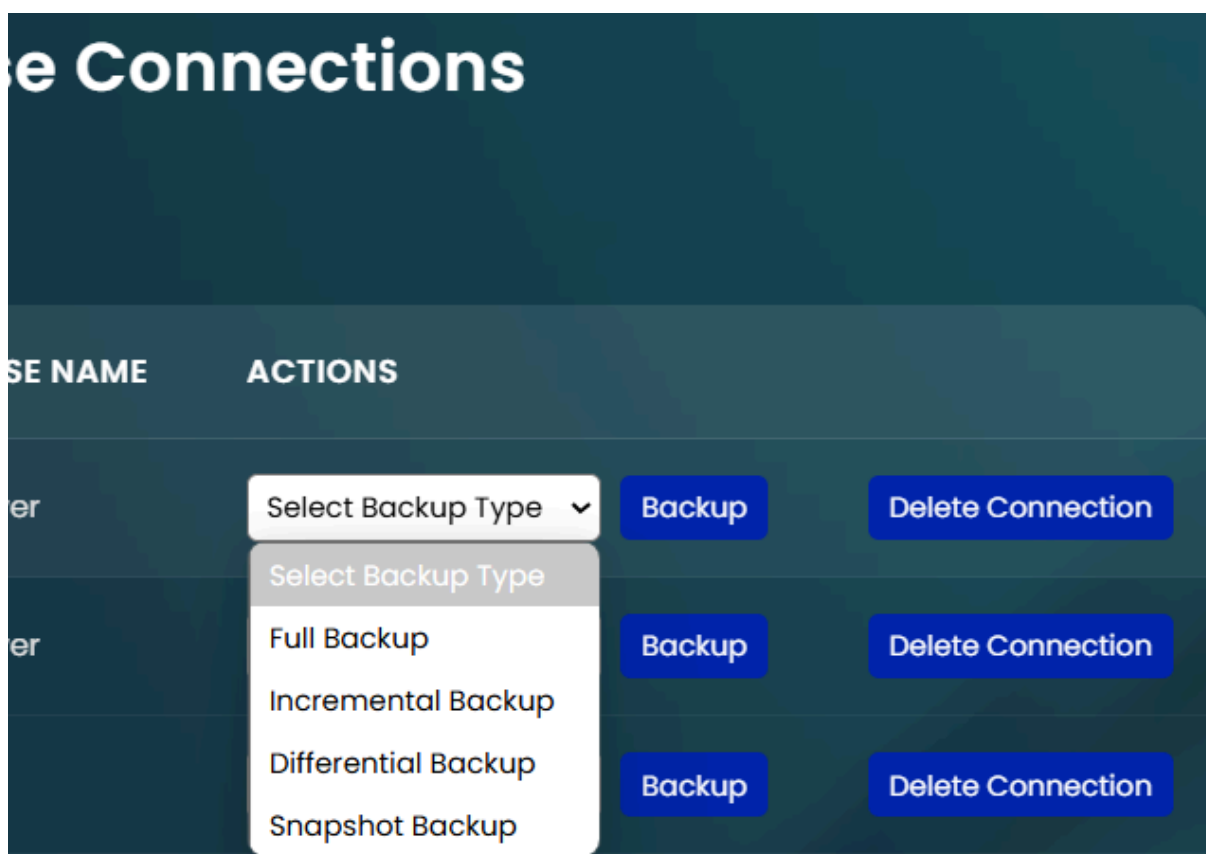
Στο σενάριο όπου ο χρήστης έχει συμπληρώσει σωστά όλα τα απαραίτητα πεδία της φόρμας δημιουργίας σύνδεσης και η δοκιμαστική σύνδεση πραγματοποιηθεί επιτυχώς, τότε εμφανίζεται με πράσινο χρώμα μήνυμα επιτυχούς σύνδεσης, με τό όνομα της εκάστοτε βάσης να εμφανίζεται ως ένδειξη ότι υπάρχει. Στην περίπτωση αυτή το κουμπί της αποθήκευσης της σύνδεσης γίνεται διαθέσιμο και ο χρήστης μπορεί πλέον να αποθηκεύσει την σύνδεση του με σιγουριά ότι τα στοιχεία αυθεντικοποίησης που έχει δώσει είναι σωστά. Στην συνέχεια μόλις ο χρήστης πατήσει το κουμπί αποθήκευσης, ενημερώνεται με notification message ότι η σύνδεση του έχει αποθηκευτεί επιτυχώς και ανακατευθύνεται στην φόρμα των συνδέσεων του.

Στην φόρμα των αποθηκευμένων συνδέσεων, ο χρήστης μπορεί να δει όλες του τις συνδέσεις, με τις λεπτομέρειες της κάθε μιας να αναγράφονται σε έναν ενιαίο πίνακα.



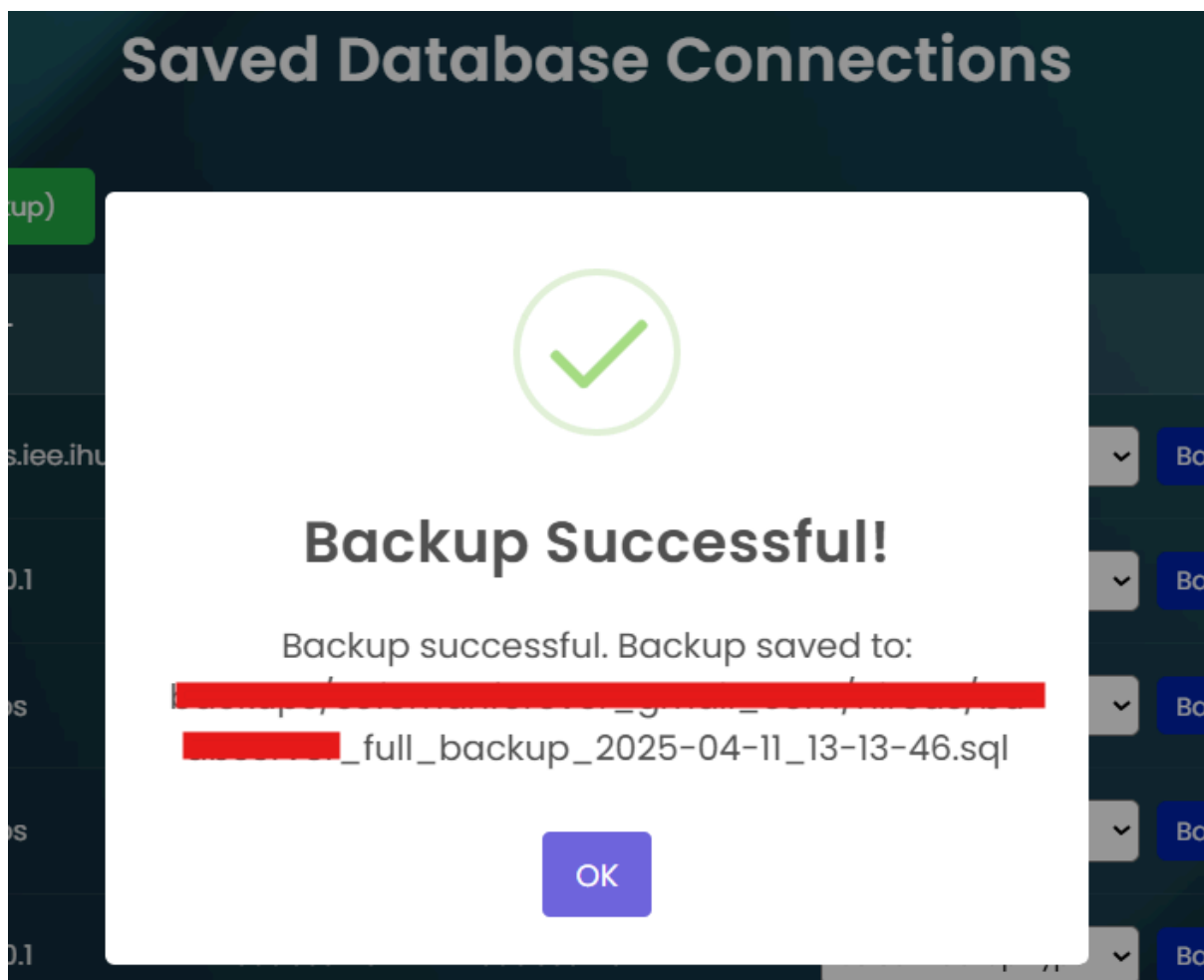
Εικόνα 5.2.11: Φόρμα αποθηκευμένων συνδέσεων χρήστη

Τα στοιχεία της κάθε σύνδεσης που εμφανίζονται είναι ο host, το username, το database name και οι επιλογές του χρήστη για το είδος του backup που επιθυμεί να πραγματοποιήσει στην εκάστοτε βάση.



Εικόνα 5.2.12: Dropdown Box επιλογών είδους αντιγράφου ασφαλείας

Ο χρήστης μπορεί να επιλέξει ανάμεσα στα 4 είδη δημιουργίας αντιγράφων ασφαλείας (Full , Incremental , Differential και Snapshot). Με το πάτημα του κουμπιού backup εκτελείται η επιλεγμένη διαδικασία και εμφανίζεται αντίστοιχο μήνυμα επιτυχίας με το όνομα του δημιουργημένου αντιγράφου.



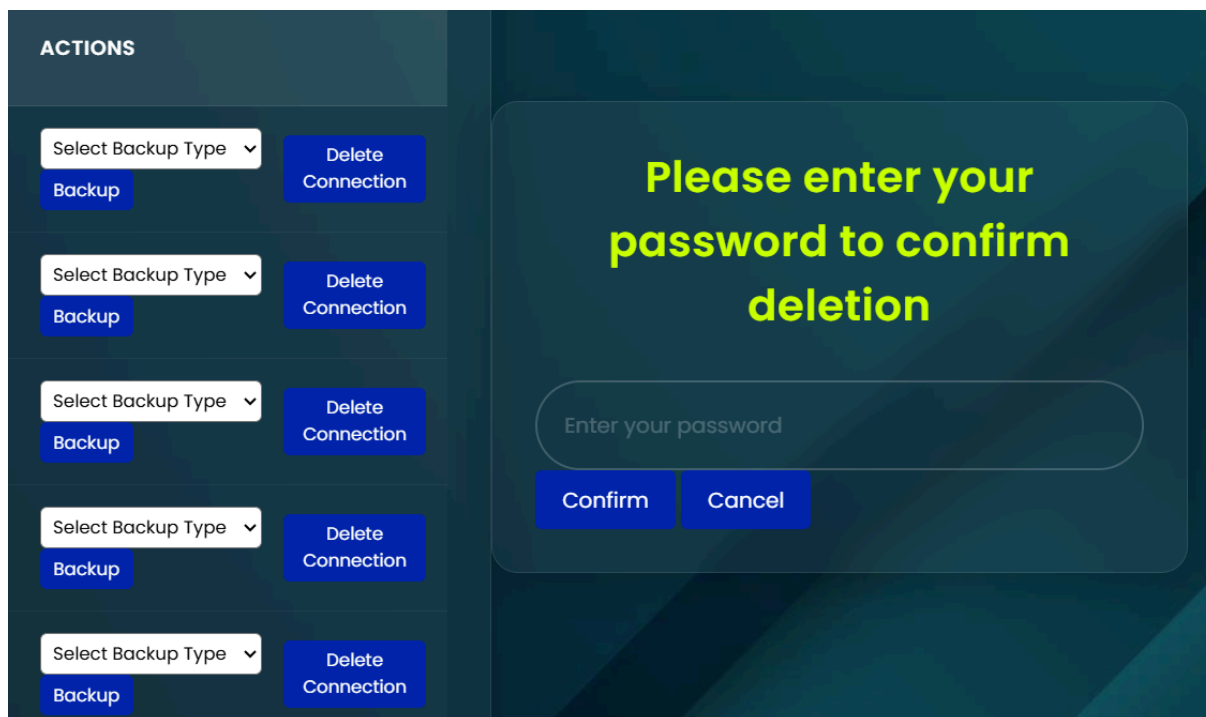
Εικόνα 5.2.13: Μήνυμα επιτυχούς δημιουργίας αντιγράφου ασφαλείας

Το αντίγραφο ασφαλείας έχει δημιουργηθεί επιτυχώς, με το όνομα του να υποδεικνύει το είδος του αλλά και την χρονική στιγμή που δημιουργήθηκε.

Ο χρήστης μπορεί να επιθυμεί να δημιουργήσει αντίγραφα ασφαλείας για όλες τις συνδέσεις του ταυτόχρονα. Με το κουμπί BACKUP ALL , όλες οι τιμές των dropdown boxes ορίζονται σε Full Backup και εκτελούνται για όλες τις συνδέσεις που έχει αποθηκευμένες ο χρήστης ταυτόχρονα, χωρίς να χρειάζεται ο χρήστης να πραγματοποιεί ατομικά για κάθε μια από αυτές Full Backups.

Ταυτόχρονα ο χρήστης μπορεί αν επιθυμεί να διαγράψει οποιαδήποτε σύνδεση δεν χρειάζεται πλέον με το κλικάρισμα του κουμπιού “Delete Connection” της αντίστοιχης σύνδεσης. Ωστόσο για την αποφυγή misclicks η οποιοδήποτε είδος ατυχήματος, πριν διαγραφεί μια σύνδεση, εμφανίζεται μια

μικρή φόρμα που ζητάει από τον χρήστη να εισάγει τον κωδικό του προκειμένου να βεβαιωθεί η εφαρμογή ότι ο χρήστης επιθυμεί να διαγράψει την εκάστοτε σύνδεση.



Εικόνα 5.2.14: Υποβολή κωδικού για διαγραφή σύνδεσης

Με την πληκτρολόγηση του κωδικού πρόσβασης, ο χρήστης πατάει confirm και εφόσον εξακριβωθεί ότι ο κωδικός που δόθηκε είναι ο σωστός, η σύνδεση διαγράφεται επιτυχώς από τον πίνακα συνδέσεων του χρήστη. Στην περίπτωση που ο χρήστης αλλάξει γνώμη και θέλει να κρατήσει την σύνδεση, πατάει το κουμπί cancel και επιστρέφει κανονικά στον πίνακα συνδέσεων.

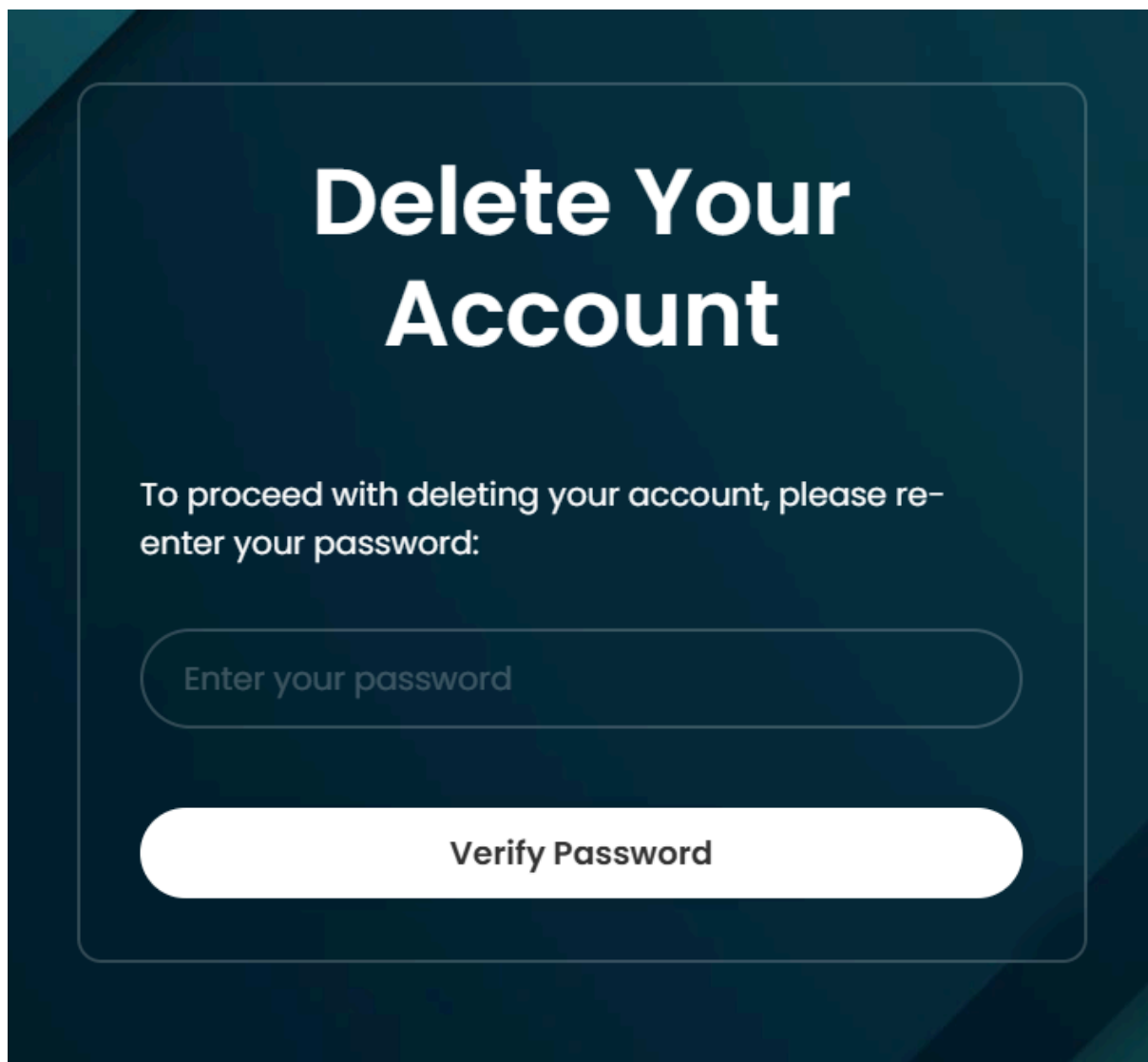
Στην συνέχεια επιλέγουμε το σύνδεσμο “My Backup Files” , προκειμένου να δούμε όλα τα δημιουργημένα μας αρχεία αντιγράφων ασφαλείας.



Εικόνα 5.2.15: Φόρμα αποθηκευμένων αντιγράφων ασφαλείας χρήστη

Χάρη στην δημιουργία αντίστοιχων directories και subdirectories για κάθε μία από τις διαφορετικές συνδέσεις του χρήστη, τα backups κατηγοριοποιούνται για ευκολότερη εύρεση τους. Ο χρήστης έχει την δυνατότητα να κατεβάσει τοπικά το sql αρχείο που δημιουργείται. Ταυτόχρονα, μπορεί αν επιθυμεί να διαγράψει οποιοδήποτε αντίγραφο ασφαλείας δεν του είναι πλέον χρήσιμο. Στην παραπάνω εικόνα παρατηρούμε ότι στην ονοματολογία του κάθε αντιγράφου ασφαλείας αναγράφεται το είδος του backup, την ημερομηνία αλλά και την ακριβή ώρα που δημιουργήθηκε.

Στην συνέχεια, αν ο χρήστης για οποιονδήποτε λόγο επιθυμεί να διαγράψει ολόκληρο το το προφίλ, υπάρχει πάνω στο navigation bar το κουμπί DELETE USER. Εννοείται πως προκειμένου να μπορέσει να διαγράψει ολόκληρο το προφίλ του χρειάζεται να ακολουθήσει 2 βήματα ασφαλείας προς αποφυγή ενεργειών που μπορεί να οδηγήσουν σε απώλειες δεδομένων εξαιτίας λαθών.



Delete Your Account

To proceed with deleting your account, please re-enter your password:

Verify Password

Εικόνα 5.2.16: Διαδικασία διαγραφής λογαριασμού (υποβολή κωδικού πρόσβασης)

Αρχικά, ο χρήστης θα πρέπει να πληκτρολογήσει τον κωδικό πρόσβασης του και εφόσον εξακριβωθεί ότι είναι σωστός, τότε μπορεί να προχωρήσει την διαδικασία διαγραφής. Στην συνέχεια καλείται να επιλέξει ρητά ότι επιθυμεί να διαγραφεί ο λογαριασμός του.

Delete Your Account

To proceed with deleting your account, please re-enter your password:

.....

Verify Password

Are you sure you want to delete your account?

- ☒ Yes, delete my account
- ☐ No, keep my account

Confirm

Εικόνα 5.2.17: Διαδικασία διαγραφής λογαριασμού (επιβεβαίωση διαγραφής)

Με την επιλογή “Yes, delete my account” , ο χρήστης έχει ολοκληρώσει και τα 2 βήματα ασφαλείας που απαιτούνται για την διαγραφή του λογαριασμού του και στην συνέχεια εμφανίζεται μήνυμα επιτυχούς διαγραφής όλων των δεδομένων του (συνδέσεις, προφίλ, αντίγραφα ασφαλείας), και ανακατευθύνεται στην φόρμα login.

Εάν ο χρήστης αλλάξει γνώμη και δεν επιθυμεί τελικά να διαγράψει το προφίλ του, επιλέγει την δεύτερη επιλογή “No, keep my account” . Στην περίπτωση αυτή η διαδικασία διαγραφής ακυρώνεται και ο χρήστης ανακατευθύνεται στην αρχική του σελίδα.

Κεφάλαιο 6ο: Συμπεράσματα και μελλοντικές επεκτάσεις

6.1 Συμπεράσματα

Η παρούσα πτυχιακή εργασία είχε ως βασικό στόχο την υλοποίηση μιας διαδικτυακής εφαρμογής που θα δίνει την δυνατότητα στους χρήστες της να διαχειρίζονται απομακρυσμένες συνδέσεις βάσεων δεδομένων και να δημιουργούν εύκολα και άμεσα αντίγραφα ασφαλείας. Με στόχο την επίτευξη αυτού του σεναρίου, δημιούργησα ένα πλήρες λειτουργικό σύστημα, το οποίο με την χρήση τεχνολογιών όπως PHP, MySQL, PostgreSQL, SSH tunneling και AJAX παρέχει στους χρήστες δυναμική και ασφαλή αλληλεπίδραση.

Συγκεκριμένα, η χρήση του SSH tunneling[48] ενίσχυσε σημαντικά την ασφάλεια των συνδέσεων, παρέχοντας έτσι την δυνατότητα για απομακρυσμένη πρόσβαση σε βάσεις δεδομένων, χωρίς την έκθεση τους στο διαδίκτυο. Παράλληλα, μέσω της χρήσης AJAX, επιτεύχθηκε η βελτίωση της

εμπειρίας χρήστη με την αποφυγή περιττών ανανεώσεων της σελίδας και την άμεση εμφάνιση δυναμικών μηνυμάτων επιβεβαίωσης ή σφαλμάτων (SweetAlert[54]).

Επιπλέον, προστέθηκαν λειτουργίες όπως διαγραφή σύνδεσης με επιβεβαίωση μέσω κωδικού χρήστη, υποστήριξη πολλαπλών τύπων αντιγράφων ασφαλείας (full, incremental, differential, snapshot), καθώς και την δυνατότητα προβολής και λήψης των αρχείων backup που δημιουργούνται ανά χρήστη και ανα βάση δεδομένων.

Η βάση δεδομένων οργανώθηκε με βασικότερο σκοπό την ασφάλεια και την σωστή καταχώρηση και αποθήκευση των στοιχείων του εκάστοτε χρήστη, με χρήση μοναδικού αναγνωριστικού το email και καταγραφή των ενεργειών (logs).

Τέλος, σημαντικό ρόλο στην διαμόρφωση του περιβάλλοντος χρήστη έπαιξε η χρήση σύγχρονων εργαλείων UI όπως SweetAlert και modals.

Συνολικά, αυτό που ήθελα περισσότερο από αυτήν την πτυχιακή είναι να αποδείξω ότι είναι εφικτή η δημιουργία ενός πλήρους συστήματος δημιουργίας και διαχείρισης αντιγράφων ασφαλείας βάσεων δεδομένων, παρέχοντας μια λειτουργική λύση για μελλοντική επέκταση σε επαγγελματικό επίπεδο.

6.2 Μελλοντικές Επεκτάσεις

Κατά την διάρκεια της πτυχιακής μου εργασίας, προσπάθησα να δημιουργήσω ένα λειτουργικό και εύχρηστο εργαλείο που να επιτρέπει τους χρήστες να διαχειρίζονται αντίγραφα ασφαλείας από τις εκάστοτε βάσεις δεδομένων στις οποίες έχουν πρόσβαση και να δημιουργούν αντίγραφα ασφαλείας από ένα ενιαίο web περιβάλλον. Αν και αυτή την στιγμή υποστηρίζονται μόνο τα πλήρη αντίγραφα (full) ασφαλείας, θεωρώ πως υπάρχουν αρκετές δυνατότητες για μελλοντική εξέλιξη της εφαρμογής.

Μια σημαντική επέκταση που θα ήθελα να προσθέσω στην εφαρμογή μου είναι η υποστήριξη και των υπολοίπων ειδών αντιγράφων ασφαλείας, καθώς αυτοί οι τύποι προσφέρουν μεγαλύτερη ευελιξία και αυξημένη εξοικονόμηση χρόνου και αποθηκευτικού χώρου. Δυστυχώς, λόγω περιορισμένων δικαιωμάτων πρόσβασης στις βάσεις δεδομένων των χρηστών, δεν κατέστη στην παρούσα φάση δυνατό να ενεργοποιηθούν αυτές οι λειτουργίες. Ωστόσο, μελλοντικά, ίσως να μπορεί να βρεθεί λύση μέσω συνεργασίας με τους διαχειριστές των εκάστοτε βάσεων η γενικότερα με οποιονδήποτε πάροχο μπορεί να προσφέρει αυτά τα απαραίτητα για την πλήρη λειτουργικότητα της εφαρμογής δικαιώματα.

Εκτός από αυτό, μελλοντικά σκοπεύω να προσθέσω και την δυνατότητα για προγραμματισμένα backups, ώστε οι χρήστες να μπορούν να ορίζουν συγκεκριμένες μέρες και ώρες για αυτόματη δημιουργία αλλά και λήψη αντιγράφων ασφαλείας. Ταυτόχρονα, θα ήταν ιδιαίτερα χρήσιμη η προσθήκη ειδοποιήσεων μέσω email, ώστε να ενημερώνονται οι χρήστες για την επιτυχία ή αποτυχία κάθε διαδικασίας δημιουργίας αντιγράφων ασφαλείας.

Τέλος θα ήθελα κάποια στιγμή μελλοντικά να επεκτείνω την υποστήριξη της εφαρμογής και σε άλλους τύπους βάσεων όπως MongoDB ή Oracle.

Αυτές είναι οι κυριότερες ιδέες που έχω για το μέλλον της εφαρμογής αυτής και ελπίζω να καταφέρω σταδιακά και με επιτυχία να τις υλοποιήσω.

ΒΙΒΛΙΟΓΡΑΦΙΑ ΠΤΥΧΙΑΚΗΣ

Βιβλία

- [36] D. Sklar and A. Trachtenberg, *PHP Cookbook*, 3rd ed., O'Reilly, 2014.
- [41] R. Lerdorf, K. Tatroe, and P. MacIntyre, *Programming PHP*, 3rd ed. Sebastopol, CA: O'Reilly Media, 2013.
- [42] B. Beighley and C. Morrison, *Head First PHP & MySQL*, 2nd ed. Sebastopol, CA: O'Reilly Media, 2008.
- [43] K. Allen, *Beginning AJAX with PHP: From Novice to Professional*, Berkeley, CA: Apress, 2006.
- [44] J. Duckett, *HTML and CSS: Design and Build Websites*, Indianapolis, IN: Wiley, 2011.
- [45] D. Flanagan, *JavaScript: The Definitive Guide*, 6th ed. Sebastopol, CA: O'Reilly Media, 2011.
- [46] B. Forta, *MySQL Crash Course*, Indianapolis, IN: Sams Publishing, 2005.
- [47] K. Douglass, *PostgreSQL: Up and Running*, 3rd ed. Sebastopol, CA: O'Reilly Media, 2017.
- [48] D. Barrett, *SSH, The Secure Shell: The Definitive Guide*, Sebastopol, CA: O'Reilly Media, 2005.
- [51] T. Powell, *Ajax: The Complete Reference*. New York: McGraw-Hill, 2008.
- [52] E. Meyer, *CSS: The Definitive Guide*, 4th ed. Sebastopol: O'Reilly Media, 2017.

Πατέντες

- [49] J. Resig and D. Bigham, "Method and System for Implementing AJAX in Web Applications," U.S. Patent 7,123,456, Oct. 17, 2006.

Data Sheet

- [50] Free Software Foundation, "mkdir: make directories" GNU Coreutils Manual <https://man7.org/linux/man-pages/man1/mkdir.1.html>

Application Note

- [54] SweetAlert2, "Frameworks Integrations – SweetAlert2" <https://sweetalert2.github.io/#frameworks-integrations>
- [55] jQuery Foundation, "jQuery API Documentation" <https://api.jquery.com/>

Internet Site

- [4] MySQL, "The mysqldump command" <https://dev.mysql.com/doc/refman/8.4/en/mysqldump.html>
- [5] PostgreSQL, "pg_dump" <https://www.postgresql.org/docs/8.1/app-pgdump.html>
- [6] DbVisualizer, "A Complete Guide to pg_dump With Examples, Tips, and Tricks" <https://www.dbvis.com/thetable/a-complete-guide-to-pg-dump-with-examples-tips-and-tricks/>

- [7] AWS, “mysqldump and mysqlpump”
<https://docs.aws.amazon.com/prescriptive-guidance/latest/migration-large-mysql-mariadb-databases/mysqldump-and-mysqlpump.html>
- [8] Red Hat, “Backing up PostgreSQL data using PostgreSQL”
https://docs.redhat.com/fr/documentation/red_hat_enterprise_linux/9/html/configuring_and_using_database_servers/backing-up-postgresql-data_using-postgresql#advantages_and_limitations_of_file_system_backing_up
- [9] Depesz, “How to effectively dump PostgreSQL databases”
<https://www.depesz.com/2019/12/10/how-to-effectively-dump-postgresql-databases/>
- [10] MySQL, “Backup types” <https://dev.mysql.com/doc/refman/8.4/en/backup-types.html>
- [12] MySQL, “mysqldump SQL format”
<https://dev.mysql.com/doc/mysql-backup-excerpt/8.0/en/mysqldump-sql-format.html>
- [13] PostgreSQL, “Backup and restore” <https://www.postgresql.org/docs/current/backup-dump.html>
- [14] ScaleGrid, “MySQL backups best practices”
<https://scalegrid.io/blog/mysql-backups-best-practices/>
- [15] IBM, “Incremental backups”
<https://www.ibm.com/docs/en/netezza?topic=nc-incremental-backups-2>
- [16] A. Talrashid, “MySQL Incremental Backup using Binary Logs”
<https://aatalrashid.medium.com/mysql-incremental-backup-using-binary-logs-bin-logging-42473adac650>
- [17] pgBackRest, “User guide” <https://pgbackrest.org/user-guide.html#concept/backup>
- [18] PostgreSQL, “Continuous archiving and point-in-time recovery”
<https://www.postgresql.org/docs/current/continuous-archiving.html>
- [19] uBackup, “MySQL Differential Backup”
<https://www.ubackup.com/enterprise-backup/mysql-differential-backup-0020.html>
- [20] PostgreSQL, “pg_basebackup” <https://www.postgresql.org/docs/current/app-pgbasebackup.html>
- [21] Rubrik, “What is a snapshot backup?”
<https://www.rubrik.com/insights/what-is-a-snapshot-backup>
- [22] Lullabot, “MySQL backups using LVM snapshots”
<https://www.lullabot.com/articles/mysql-backups-using-lvm-snapshots>
- [23] dbnapper, “How to create a PostgreSQL database snapshot”
<https://dbnapper.com/blog/how-to-create-a-postgresql-database-snapshot>
- [24] MDN Web Docs, “HTML” <https://developer.mozilla.org/en-US/docs/Web/HTML>
- [29] W3Docs, “HTML styles” <https://www.w3docs.com/learn-html/html-styles.html>
- [32] W3Schools, “PHP Introduction” https://www.w3schools.com/php/php_intro.asp
- [35] W3Schools, “PHP Error Handling” https://www.w3schools.com/php/php_error.asp

[38] Learn to Code With Me, “The Ultimate SQL Guide for Beginners”
<https://learntocodewith.me/posts/sql-guide/>

[39] Astera, “SQL vs NoSQL – Differences & Comparison”
<https://www.astera.com/knowledge-center/sql-vs-nosql/>

[53] Postman, “What is API Testing? A Guide to Testing APIs”
<https://www.postman.com/api-platform/api-testing/>

Journal Articles

[1] E. Kriti, “Ψηφιακή διαχείριση εγγράφων: Γιατί πρέπει να αρνηθείτε τα έντυπα έγγραφα,” *eKriti*,
<https://www.ekriti.gr/business/psifiaki-diaheirisi-eggrafon-giati-prepei-na-arnitheite-ta-entypa-eggrafa>

[2] Zisopoulos, “Ψηφιακός μετασχηματισμός: Πλεονεκτήματα και μειονεκτήματα,” *Zisopoulos*,
<https://www.zisopoulos.com/blog/124-psifiakos-metaskhimatismos-pleonektimata-kai-meionektimata>

[3] Zisopoulos, “Προχωρώντας σε ένα paperless χωρίς χαρτί γραφείο,” *Zisopoulos*,
<https://www.zisopoulos.com/blog/60-prokhorwntas-se-ena-paperless-khoris-kharti-ghrafio>

[11] M. Chelongar, “PostgreSQL backup types comparison & contrast,” *Medium*,
<https://medium.com/@masoud.chelongar/postgresql-backup-types-comparison-contrast-b4945177cab>

[25] Unstop, “Πλεονεκτήματα και μειονεκτήματα του HTML,” *Unstop*,
<https://unstop.com/blog/advantages-and-disadvantages-of-html>

[26] GetResponse, “Συμβουλές και εργαλεία για τη συγγραφή SEO-friendly περιεχομένου,” *GetResponse*,
<https://www.getresponse.com/blog/tips-and-tools-to-write-seo-friendly-content>

[27] GeeksforGeeks, “Πλεονεκτήματα και μειονεκτήματα του HTML,” *GeeksforGeeks*,
<https://www.geeksforgeeks.org/advantages-and-disadvantages-of-html/>

[28] Mozilla, “Χρήση Service Workers στο Web API,” *Mozilla Developer Network*,
https://developer.mozilla.org/en-US/docs/Web/API/Service_Worker_API/Using_Service_Workers

[30] PrimerCSS, “Τι είναι το εξωτερικό CSS;,” *PrimerCSS*, <https://primercss.io/what-is-external-css/>

[31] GeeksforGeeks, “Πλεονεκτήματα και μειονεκτήματα του CSS,” *GeeksforGeeks*,
<https://www.geeksforgeeks.org/advantages-and-disadvantages-of-css/>

[33] GeeksforGeeks, “Πλεονεκτήματα και μειονεκτήματα του PHP,” *GeeksforGeeks*,
<https://www.geeksforgeeks.org/advantages-and-disadvantages-of-php/>

[34] CustomLab, “Πλεονεκτήματα και μειονεκτήματα του PHP,” *CustomLab*,
<https://customlab.gr/php-advantages-disadvantages/>

[37] ProjectManagers, “Top 10 Pros and Cons of Using PHP,” *ProjectManagers*,
<https://projectmanagers.net/top-10-pros-and-cons-of-using-php/>

[40] GeeksforGeeks, “Πλεονεκτήματα και μειονεκτήματα του SQL vs NoSQL βάσεων δεδομένων,” *GeeksforGeeks*,
<https://www.geeksforgeeks.org/what-are-the-advantages-and-disadvantages-of-using-sql-vs-nosql-data-bases/>