

# ΙΟΝΙΟ ΠΑΝΕΠΙΣΤΗΜΙΟ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ



- Πτυχιακή Εργασία -

Ανάπτυξη διαδικτυακής εφαρμογής για την υλοποίηση ψηφιακής ντουλάπας (Digital Wardrobe)

Ιάκωβος Ιάσων Τούντας Π2019034

Επιβλέπων: - Αριστείδης Βραχάτης -

11 Νοεμβρίου 2024

## Επιβλέπων

Αριστείδης Βραχάτης, Επίκουρος Καθηγητής,

Ιόνιο Πανεπιστήμιο

## Τριμελής Επιτροπή

Αριστείδης Βραχάτης, Επίκουρος Καθηγητής,

Ιόνιο Πανεπιστήμιο

Θεμιστοκλής Έξαρχος, Αναπληρωτής Καθηγητής,

Ιόνιο Πανεπιστήμιο

Ανδρέας Καναβός, Αναπληρωτής Καθηγητής,

Ιόνιο Πανεπιστήμιο

## ΠΕΡΙΛΗΨΗ

Η κατασκευή και ανάπτυξη ιστοσελίδων και διαδικτυακών εφαρμογών είναι η διαδικασία αναπαράστασης πολυμεσικού περιεχομένου ή υπερκειμένου, οι οποίες μέσω του διαδικτύου μπορούν και προβάλλονται στους χρήστες που αλληλεπιδρούν μαζί τους κυρίως χάρη στα προγράμματα περιήγησης (λ.χ. Google Chrome). Κάποιους από τους σημαντικότερους τομείς σχεδιασμού ιστοσελίδων και διαδικτυακών εφαρμογών αποτελούν ο σχεδιασμός γραφικών, ο σχεδιασμός διεπαφής και εμπειρίας χρήστη και η βελτιστοποίηση μηχανών αναζήτησης, αντικείμενα που συχνά αναπτύσσονται από πολλά άτομα σε ομάδες.

Αυτές οι ομάδες διαχρονικά είχαν ως κύριο σκοπό την επίλυση ανθρώπινων προβλημάτων προκειμένου να κάνουν τη ζωή πιο απλή, οργανωμένη και τις δουλειές πιο γρήγορες. Στο παρελθόν οι ιστοσελίδες έλυσαν προβλήματα όπως η πρόσβαση στην πληροφορία, η επικοινωνία και ανάπτυξη μιας επιχείρησης ηλεκτρονικού εμπορίου. Σήμερα, πέρα από όσα αναφέρθηκαν προηγουμένως, συνδέουν τους ανθρώπους μέσω των κοινωνικών δικτύων, παρέχουν σημαντικές δυνατότητες όπως η απομακρυσμένη εργασία και οι ηλεκτρονικές αγορές, ενισχύοντας έτσι την καθημερινή ζωή. Μελλοντικά, χάρη στην τεχνητή νοημοσύνη πολλά σύγχρονα προβλήματα θα απλοποιηθούν ή και θα απαλειφθούν τελείως.

Ωστόσο, στη σημερινή εποχή ένα πρόβλημα που δύσκολα μπορεί να διαχειριστεί ένας άνθρωπος, ανεξαρτήτως πόσα βήματα μπροστά κάνει συνεχώς η τεχνολογία, είναι η οργάνωση και τάξη των πραγμάτων είτε στην επαγγελματική είτε στην προσωπική ζωή. Αυτό το πρόβλημα πηγάζει από συμπεριφορές όπως η τεμπελιά αλλά και καταστάσεις όπως κακή ψυχική υγεία ή χαμηλή αυτοεκτίμηση. Εμείς θα εστιάσουμε στην αδυναμία να οργανώσει την ντουλάπα με τα ρούχα του, πράγμα που πολλές φορές οδηγεί σε ακατάστατους χώρους, άγχος ή/και άσκοπα έξοδα καθώς το ανθρώπινο μυαλό τείνει να ξεχνάει τα αντικείμενα που δεν βλέπει συχνά το μάτι με αποτέλεσμα να αγοράζει και άλλα.

Στον παρόν κείμενο, θα παρουσιάσουμε ένα ολοκληρωμένο εργαλείο σε μορφή διαδικτυακής εφαρμογής με όνομα “Ψηφιακή Ντουλάπα” (Digital Wardrobe) που λύνει το παραπάνω πρόβλημα, καθώς με τις λειτουργίες του προσφέρει ευκολία και ταχύτητα στην οργάνωση της γκαρνταρόμπας, και εξοικονόμηση χρόνου και χρήματος των χρηστών. Για την υλοποίησή του χρησιμοποιήθηκαν βασικές τεχνολογίες όπως η MongoDB για την βάση δεδομένων, το Node.js και Express.js για το Back-End, και το React.js για το Front-End.

**Λέξεις κλειδιά:** Ψηφιακή Ντουλάπα, Οργάνωση, Διαδικτυακή Εφαρμογή, Ασφάλεια Δεδομένων, Ένδυση

# ABSTRACT

The design and development of websites and web applications is the process of multimedia representation or hypertext content, which can be displayed on the internet to the user who then interacts with them through browsers (e.g. Google Chrome, Mozilla Firefox). Some of the most important areas of websites and web applications design are graphic design, user interface (UI) and user experience (UX) design and search engine optimization (SEO), often developed by several people in groups.

Throughout time, these groups had the main purpose of solving human problems in order to make life simpler, organized and jobs faster. In the past, websites solved problems like access to information, communication, and the development of a commercial e-commerce business. Today, apart from what was mentioned earlier, they connect people through social networks, provide important features such as remote working and online shopping, thus enhancing everyday life. In the future, thanks to artificial intelligence (AI) many modern problems will be simplified or even erased completely.

However, in today's world, one problem that is hard for a human to manage, despite how many steps ahead is technology constantly making, is the organization and order of things either in professional or personal life. This problem stems from behaviors such as laziness, but also situations like poor mental health or low self-esteem. We will focus on the inability to organize one's clothing wardrobe, which many times lead to messy spaces, stress or/and unnecessary expenses as the human mind tends to forget the items that the eye does not usually see and as a result he buys more.

In this thesis, we will present a comprehensive tool in the form of a web application named "Digital Wardrobe" which solves the problem mentioned above, as its functions offer ease and speed in the organization of the wardrobe, and save time and money of users. We used key technologies for its implementation such as MongoDB for the database, Node.js and Express.js for the Back-End, and React.js for the Front-End.

**Keywords:** Digital Wardrobe, Organization, Web Application, Data Security, Clothing

## ΕΥΧΑΡΙΣΤΙΕΣ

Θα ήθελα να εκφράσω τις θερμές ευχαριστίες μου στην τριμελή επιτροπή που αποτελείται από τους καθηγητές μου κ. Αριστείδης Βραχάτης, κ. Θεμιστοκλής Έξαρχος και κ. Ανδρέας Καναβός, για την εξέταση και αξιολόγηση της παρούσας πτυχιακής εργασίας. Παράλληλα θα ήθελα εξίσου να ευχαριστήσω τον κ. Γεώργιο Δημητρακόπουλο, καθηγητή μου, για την πολύτιμη βοήθεια και καθοδήγησή του στην ολοκλήρωση της παρούσας πτυχιακής εργασίας. Η συμβολή του στην επιμέλεια και διόρθωση του περιεχομένου του κειμένου, καθώς και η υποστήριξή του στην αντιμετώπιση των προκλήσεων και τεχνικών εμποδίων που προέκυψαν κατά τη διάρκεια της υλοποίησης της διαδικτυακής εφαρμογής, υπήρξε καθοριστική.

Θα ήθελα επίσης να ευχαριστήσω την οικογένειά μου για την στήριξή τους καθόλη τη διάρκεια των σπουδών μου στο τμήμα Πληροφορικής του Ιονίου Πανεπιστημίου, και για την ενθάρρυνση και υπομονή που έδειξαν σε κάθε βήμα αυτής της προσπάθειας.

Με εκτίμηση,

Τούντας Ιάκωβος Ιάσων

# ΠΕΡΙΕΧΟΜΕΝΑ

<b>ΠΕΡΙΛΗΨΗ.....</b>	<b>3</b>
<b>ABSTRACT.....</b>	<b>4</b>
<b>ΕΥΧΑΡΙΣΤΙΕΣ.....</b>	<b>5</b>
<b>ΠΕΡΙΕΧΟΜΕΝΑ.....</b>	<b>6</b>
<b>ΚΑΤΑΛΟΓΟΣ ΕΙΚΟΝΩΝ.....</b>	<b>9</b>
<b>ΣΥΝΤΜΗΣΕΙΣ.....</b>	<b>11</b>
<b>ΓΛΩΣΣΑΡΙ ΕΣΕΝΙΚΩΝ ΟΡΩΝ.....</b>	<b>12</b>
<b>1 ΕΙΣΑΓΩΓΗ.....</b>	<b>1</b>
1.1 Παρουσίαση του Προβλήματος.....	1
1.1.1 Προβλήματα Οργάνωσης Ειδών Ένδυσης.....	1
1.1.2 Επιπτώσεις στην Καθημερινή Ζωή.....	2
1.1.3 Ανάγκη για Ψηφιακή Λύση.....	3
1.2 Σκοπός της Εφαρμογής.....	4
1.2.1 Βασικός Σκοπός.....	4
1.2.2 Βελτίωση Καθημερινότητας.....	5
1.3 Στόχοι και Προσδοκίες.....	6
1.3.1 Στόχοι.....	6
1.3.2 Προσδοκίες.....	7
1.4 Δομή της εφαρμογής.....	8
1.4.1 Back-End.....	9
1.4.2 Front-End.....	9
<b>2 BACK-END.....</b>	<b>11</b>
2.1 Εισαγωγή στο Back-End.....	11
2.1.1 Ρόλος στην εφαρμογή.....	11
2.1.2 Επικοινωνία με το Front-End.....	12
2.2 Αρχιτεκτονική και Σχεδιασμός.....	12
2.2.1 Λόγος Σχεδιασμού.....	13
2.2.2 Διαχείριση Δεδομένων και Απόκρισης.....	13
2.3 Τεχνολογίες και Εργαλεία.....	13
2.3.1 JavaScript.....	14
2.3.2 Node.js.....	14
2.3.3 Express.js.....	15
2.3.4 Βάση Δεδομένων.....	15
2.3.4.1 MongoDB.....	16
2.3.4.2 Mongoose.....	17
2.3.5 Άλλες Βιβλιοθήκες.....	19
2.3.5.1 Multer.....	19
2.3.5.2 JWT (jsonwebtoken).....	20
2.3.5.3 bcrypt.....	21
2.3.5.4 Helmet.....	22
2.3.5.5 Morgan.....	22
2.3.5.6 CORS.....	22

2.3.5.7 Router.....	23
2.3.5.8 dotenv.....	24
2.3.5.9 body-parser.....	25
2.3.5.10 express-validator.....	25
2.3.5.11 Path.....	26
2.4 Υλοποίηση και Κώδικας.....	26
2.4.1 Προετοιμασία Περιβάλλοντος Ανάπτυξης.....	27
2.4.2 Ανάπτυξη και Δημιουργία Μοντέλων.....	27
2.4.3 Διαχείριση Εικόνων και Ανάπτυξη Διαδρομών.....	28
2.4.4 Δημιουργία και Διαχείριση Αυθεντικοποίησης.....	29
2.4.5 Έλεγχος, δοκιμή και προσαρμογές για το Front-End.....	31
2.4.6 Εργαλεία Υποστήριξης.....	32
2.5 Προκλήσεις και Εμπόδια.....	32
2.5.1 Προκλήσεις και Λύσεις.....	33
2.5.2 Τεχνικά Εμπόδια και Λύσεις.....	33
2.6 Συμπεράσματα.....	34
<b>3 FRONT-END.....</b>	<b>36</b>
3.1 Εισαγωγή στο Front-End.....	36
3.1.1 Ρόλος στην εφαρμογή.....	36
3.1.2 Επικοινωνία με το Back-End και τη MongoDB.....	37
3.2 Αρχιτεκτονική και Σχεδιασμός.....	37
3.2.1 Λόγος Σχεδιασμού.....	37
3.2.2 Ευχρηστία και Προσβασιμότητα.....	38
3.3 Τεχνολογίες και Εργαλεία.....	38
3.3.1 HTML.....	39
3.3.2 CSS.....	39
3.3.3 JavaScript.....	40
3.3.4 React.js.....	41
3.3.5 Axios.....	41
3.4 Υλοποίηση Διεπαφής και Εμπειρίας Χρήστη.....	43
3.4.1 Αρχικός Σχεδιασμός και Στήσιμο της Εφαρμογής.....	43
3.4.2 Σύνδεση, Εγγραφή και Προφίλ Χρηστών.....	44
3.4.3 Αρχική, Επικεφαλίδα & Υποσέλιδο: Δομή και Λειτουργίες.....	45
3.4.4 Βασικές Λειτουργίες.....	47
3.4.4.1 Ρούχα.....	47
3.4.4.2 Σύνολα.....	49
3.4.4.3 Περιστάσεις.....	51
3.4.4.4 Ταξιδιωτικές Λίστες.....	53
3.4.5 Στιλιστικές Επιλογές.....	55
3.4.6 Δοκιμή και Έλεγχος.....	57
3.5 Προκλήσεις και Τεχνικά Εμπόδια.....	57
3.5.1 Προκλήσεις και Λύσεις.....	57
3.5.2 Τεχνικά Εμπόδια και Λύσεις.....	58
3.6 Συμπεράσματα.....	59

<b>4 ΣΥΖΗΤΗΣΗ, ΒΕΛΤΙΩΣΕΙΣ ΚΑΙ ΣΥΜΠΕΡΑΣΜΑΤΑ.....</b>	<b>61</b>
4.1 Συζήτηση.....	61
4.2 Βελτιώσεις.....	62
4.3 Τελικά Συμπεράσματα.....	66
<b>ΒΙΒΛΙΟΓΡΑΦΙΑ.....</b>	<b>68</b>

# ΚΑΤΑΛΟΓΟΣ ΕΙΚΟΝΩΝ

<b>Εικόνα 1:</b> Η επικοινωνία του Front-End, Back-End και μιας βάσης δεδομένων (από εδώ)....	<b>12</b>
<b>Εικόνα 2:</b> Οι συλλογές της βάσης δεδομένων του “Digital Wardrobe” όπως φαίνονται στην εφαρμογή MongoDB Compass.....	<b>16</b>
<b>Εικόνα 3:</b> Διάγραμμα σχήματος βάσης δεδομένων του “Digital Wardrobe” (δημιουργήθηκε στο draw.io).....	<b>17</b>
<b>Εικόνα 4:</b> Τμήμα κώδικα στο αρχείο app.js που συνδέει το Back-End με τη βάση δεδομένων MongoDB.....	<b>18</b>
<b>Εικόνα 5:</b> Κώδικας του μοντέλου των ρούχων με όλα τα πεδία του (Στιγμιότυπο από το αρχείο clothes.js στην εφαρμογή Visual Studio Code).....	<b>19</b>
<b>Εικόνα 6:</b> Παράδειγμα τμήματος κώδικα χρήσης του Multer στο αρχείο διαδρομών των χρηστών (users_routes.js).....	<b>20</b>
<b>Εικόνα 7:</b> Διαδρομή για σύνδεση των χρηστών, όπου μετά την επαλήθευσή τους, δημιουργούνται ένα access και ένα refresh token και αποστέλλονται σε αυτούς (κώδικας από auth_routes.js).....	<b>21</b>
<b>Εικόνα 8:</b> Κύριος κώδικας χρήσης CORS στο αρχείο app.js του Back-End της εφαρμογής....	<b>23</b>
<b>Εικόνα 9:</b> Τμήμα κώδικα που επιτρέπει στους ανθεντικοποιημένους χρήστες να διαγράφουν τον λογαριασμό τους από τη βάση δεδομένων με τη μέθοδο DELETE (από το αρχείο user_routes.js).....	<b>24</b>
<b>Εικόνα 10:</b> Τμήμα κώδικα του app.js όπου διακρίνεται η χρήση του Path για την εξυπηρέτηση στατικών αρχείων από τα uploads/pfp και uploads/clothes.....	<b>26</b>
<b>Εικόνα 11:</b> Το μοντέλο των χρηστών στο Back-End με όλα τα απαραίτητα πεδία και τα προαιρετικά (αρχείο “users.js” στον φάκελο models).....	<b>27</b>
<b>Εικόνα 12:</b> Μέθοδος POST για τη διαδρομή των χρηστών με όλα τα πεδία συμπλήρωσης και σημαντικές βιβλιοθήκες (από το αρχείο user_routes.js).....	<b>29</b>
<b>Εικόνα 13:</b> Συναρτήσεις δημιουργίας των δυο tokens στο αρχείο token.js.....	<b>30</b>
<b>Εικόνα 14:</b> Διαδρομή σύνδεσης στην εφαρμογή που πραγματοποιείται στο αρχείο διαδρομής ανθεντικοποίησης και εγγραφής των χρηστών (auth_routes.js).....	<b>31</b>
<b>Εικόνα 15:</b> Αποστολή αιτήματος POST για την σύνδεση του χρήστη στη διαδικτυακή εφαρμογή (τμήμα κώδικα από το αρχείο Signin.js).....	<b>42</b>
<b>Εικόνα 16:</b> Αποστολή αιτήματος GET για την ανάκτηση όλων των ρούχων του χρήστη και την προβολή τους στη σελίδα (τμήμα κώδικα από το αρχείο Clothes.js).....	<b>43</b>
<b>Εικόνα 17:</b> Φόρμα εγγραφής χρηστών στο Digital Wardrobe, με μήνυμα καλωσορίσματος και	

καθοδήγηση στο <i>About Us</i> .....	44
<b>Εικόνα 18:</b> Καρτέλα επεξεργασίας προφίλ με προσυμπληρωμένα στοιχεία χρήστη και πλήκτρα “Update Profile” και “Cancel”.....	45
<b>Εικόνα 19:</b> Η οπική ενός συνδεδεμένου χρήστη από την αρχική του “Digital Wardrobe” μαζί με την επικεφαλίδα.....	46
<b>Εικόνα 20:</b> Υποσέλιδο της εφαρμογής με όλες τις ενότητες και τις διαθέσιμες επιλογές τους... ..	47
<b>Εικόνα 21:</b> Τρόπος εμφάνισης των ανεβασμένων ρούχων ενός χρήστη στο προφίλ του. Το ενδιάμεσο ρούχο έπειτα από κλικ εμφανίζεται περισσότερες λεπτομέρειες.....	48
<b>Εικόνα 22:</b> Dropdown των φίλτρων αναζήτησης των ρούχων με το πάτημα του κοντριού “Search Filters”.....	48
<b>Εικόνα 23:</b> Τρόπος εμφάνισης των δημιουργημένων συνόλων ενός χρήστη. Στο ενδιάμεσο σύνολο έπειτα από κλικ στον τίτλο “Cold Winter” εμφανίζεται η εικόνα του ρούχου.....	50
<b>Εικόνα 24:</b> Φόρμα δημιουργίας συνόλου με τα πεδία συμπλήρωσης, τα φίλτρα αναζήτησης ρούχου και ένα μαρκαρισμένο επιλεγμένο ρούχο ( <i>Gray Hoodie</i> ).....	51
<b>Εικόνα 25:</b> Τρόπος εμφάνισης των περιστάσεων ενός χρήστη. Στην ενδιάμεση περίσταση έπειτα από κλικ στο σύνολο με όνομα “Cold Winter” εμφανίζονται τα περιεχόμενα ρούχα.....	52
<b>Εικόνα 26:</b> Φόρμα δημιουργίας περίστασης με τα πεδία συμπλήρωσης, τα φίλτρα αναζήτησης συνόλου και ένα μαρκαρισμένο επιλεγμένο σύνολο ( <i>Neon Man</i> ).....	53
<b>Εικόνα 27:</b> Τρόπος εμφάνισης των ταξιδιωτικών λιστών ενός χρήστη. Στην ενδιάμεση ταξιδιωτική λίστα έπειτα από κλικ στο ρούχο με όνομα “Nike Jordan I Khakhi” εμφανίζεται η εικόνα του.....	54
<b>Εικόνα 28:</b> Φόρμα δημιουργίας ταξιδιωτικής λίστας με τα πεδία συμπλήρωσης, τα φίλτρα αναζήτησης ρούχων και δύο μαρκαρισμένα επιλεγμένα αντικείμενα.....	55

# ΣΥΝΤΜΗΣΕΙΣ

ADHD	Attention Deficit Hyperactivity Disorder	Διαταραχή Ελλειμματικής Προσοχής/Υπερκινητικότητας
AI	Artificial Intelligence	Τεχνητή Νοημοσύνη
AR	Augmented Reality	Επαυξημένη Πραγματικότητα
CORS	Cross-Origin Resource Sharing	Κοινή Χρήση Πόρων Διαφορετικής Προέλευσης
CSS	Cascading Style Sheets	Διαδοχικά Φύλλα Στυλ
FAQ	Frequently Asked Questions	Συχνές ερωτήσεις
HTML	HyperText Markup Language	Γλώσσα Υπερκειμένου
HTTP	Hypertext Transfer Protocol	Πρωτόκολλο Μεταφοράς Υπερκειμένου
JSON	JavaScript Object Notation	Σημειογραφία Αντικειμένου JavaScript
JS	JavaScript	- (γλώσσα προγραμματισμού)
JWT	JSON Web Token	- (βιβλιοθήκη)
npm	Node Package Manager	Διαχειριστής Πακέτων για το Node.js
PTSD	Post-Traumatic Stress Disorder	Διαταραχή Μετατραυματικού Στρες
QR	Quick Response	Κωδικός Γρήγορης Ανταπόκρισης
SQL	Structured Query Language	Δομημένη Γλώσσα Ερωτημάτων
UI	User Interface	Διεπαφή Χρήστη
UX	User Experience	Εμπειρία Χρήστη

# ΓΛΩΣΣΑΡΙ ΕΕΝΙΚΩΝ ΟΡΩΝ

About Us	Σχετικά με Εμάς
async/await	Ασύγχρονη λειτουργία/αναμονή
Back-End	Πίσω μέρος της εφαρμογής
Barcode	Γραμμωτός κώδικας
body-parser	Αναλυτής σώματος (για αιτήματα HTTP)
Clothes	Ρούχα
Console	Κονσόλα
Contact Us	Επικοινωνήστε Μαζί Μας
Create	Δημιουργία
Delete	Διαγραφή
Digital Wardrobe	Ψηφιακή Ντουλάπα
Edit	Επεξεργασία
express-validator	Επικυρωτής express (για έλεγχο δεδομένων)
Footer	Υποσέλιδο
Front-End	Μπροστινό μέρος της εφαρμογής
GET, POST, PUT, DELETE	Ανάκτηση, Αποστολή Ενημέρωση, Διαγραφή
Header	Επικεφαλίδα

Helmet	Κράνος (βιβλιοθήκη)
Help Center	Κέντρο Βοήθειας
My Account	Ο Λογαριασμός Μου
Network	Δίκτυο
Newsletter	Ενημερωτικό Δελτίο
NoSQL	Βάση δεδομένων χωρίς SQL
Occasions	Περιστάσεις
Outfits	Σύνολα
Path	Διαδρομή (βιβλιοθήκη)
require	Απαίτηση (χρήση σε κώδικα για φόρτωση βιβλιοθηκών)
required	Υποχρεωτικό
Router	Δρομολογητής
Sign In	Είσοδος
Sign Out	Έξοδος
Sign Up	Εγγραφή
Travel Lists	Ταξιδιωτικές Λίστες
Update	Ενημέρωση

# 1 ΕΙΣΑΓΩΓΗ

Σε αυτό το κεφάλαιο θα εστιάσουμε στην ιδέα πίσω από τη διαδικτυακή εφαρμογή “Digital Wardrobe”. Συγκεκριμένα, θα παρουσιαστεί τα προβλήματα τα οποία έρχεται να λύσει η εφαρμογή, τον σκοπό για τον οποίο αναπτύχθηκε και τα πλεονεκτήματα που φέρει η χρήση της καθώς και τους στόχους και τις προσδοκίες που έχουμε για την εφαρμογή. Τέλος, θα γίνει μια σύντομη περιγραφή των κεφαλαίων που ακολουθούν όπου και θα αναλυθεί ο τρόπος ανάπτυξης και σχεδιασμού της.

## 1.1 Παρουσίαση του Προβλήματος

Η ιδέα της ψηφιακής ντουλάπας προέρχεται από ένα σύνηθες πρόβλημα που αντιμετωπίζουν πολλοί άνθρωποι διαχρονικά. Αυτό το πρόβλημα αφορά τη δυσκολία οργάνωσης και συγκεκριμένα αυτή των ειδών ένδυσης σε μια ντουλάπα, πράγμα που φέρει επιπτώσεις στην καθημερινή ζωή, τόσο στην ψυχική υγεία όσο και στα οικονομικά ενός ανθρώπου που το αντιμετωπίζει [1]. Στη σήμερον ημέρα, ο άνθρωπος τείνει να ψάχνει τις λύσεις στα προβλήματά του βασιζόμενος στην τεχνολογία και σε αυτή την κίνηση βασίζεται την ιδέα μας [2]. Για την περαιτέρω εξήγηση του προβλήματος, παρακάτω θα αναλυθούν όσα προαναφέρθηκαν:

### 1.1.1 Προβλήματα Οργάνωσης Ειδών Ένδυσης

Ένα μεγάλο ποσοστό ανθρώπων στις μέρες μας αδυνατεί να θέσει σε τάξη πολλά αντικείμενα που αφορούν τη ζωή τους είτε επαγγελματικά είτε προσωπικά. Αυτά τα αντικείμενα έχουν πολλές μορφές, όπως για παράδειγμα η οργάνωση και ταξινόμηση βιβλίων σε μια βιβλιοθήκη σπιτιού ή η ορθή και μεθοδική οργάνωση πλάνου ολοκλήρωσης ενός έργου (όπως η ανάπτυξη μιας εφαρμογής) για μια εταιρία [3], [4]. Η αδυναμία αυτή μπορεί να οφείλεται σε προβλήματα στην ψυχική και σωματική υγεία.

Εμείς θα εστιάσουμε στην αδυναμία οργάνωσης ειδών ένδυσης σε μια ντουλάπα ενός σπιτιού. Αυτή η αδυναμία μπορεί να οφείλεται σε κακές συνήθειες όπως η τεμπελιά που συναντάται στους περισσότερους ανθρώπους τα τελευταία χρόνια αλλά και σε πιο σοβαρά προβλήματα όπως το ΔΕΠΥ (Διαταραχή Ελλειμματικής Προσοχής/Υπερκινητικότητας - ADHD) ή την κακή ψυχική και πνευματική υγεία [3].

Το ΔΕΠΥ είναι μια από τις συχνότερες νευροβιολογικές διαταραχές η οποία εμφανίζεται στην παιδική ηλικία και συνεχίζεται στην ενήλικη ζωή. Χαρακτηριστικά συμπτώματα αποτελούν η διάσπαση προσοχής, η παρορμητικότητα και η υπερκινητικότητα, που μπορούν να κάνουν την οργάνωση μιας ντουλάπας εξαιρετικά δύσκολη λόγω της έλλειψης εστίασης και διαχείρισης χρόνου [5], [6]. Αυτό μπορεί να οδηγήσει στη χρόνια ακαταστασία της.

Η κακή σωματική και ψυχική υγεία που οδηγεί στην αδυναμία οργάνωσης μιας ντουλάπας μπορεί να οφείλεται σε διάφορους λόγους. Η χαμηλή αυτοεκτίμηση που αποτελεί έναν από τους σημαντικότερους λόγους, είναι μια ψυχολογική έννοια που ενσωματώνει τη συνολική αξιολόγηση και αντίληψη ενός ατόμου για τον εαυτό του. Με άλλα λόγια είναι το συναίσθημα του να μην αισθάνεται κάποιος αρκετά καλός/ικανός με αποτέλεσμα να μην προσπαθεί να οργανώσει τη ζωή του αφού θεωρεί πως δεν του αξίζουν καλά πράγματα. Η κατάθλιψη επίσης παίζει έναν πολύ σημαντικό ρόλο διότι σε αυτή περιπλέκονται η έλλειψη ενέργειας και διαύγειας για τη διαχείριση πραγμάτων, κάτι που επίσης σχετίζεται άμεσα με τα προβλήματα ύπνου [7]. Παράλληλα, το άγχος συμβάλλει με τον ίδιο τρόπο, δημιουργώντας ένα νέφος στο μυαλό του ανθρώπου με αποτέλεσμα να δυσκολεύεται να συγκεντρωθεί στην οργάνωση και να τα παρατάει [2]. Τέλος, το PTSD (Διαταραχή μετατραυματικού στρες), διαταραχή που εμφανίζεται σε διάφορες ηλικίες έπειτα από σοβαρό ψυχολογικό τραύμα, μπορεί αντίστοιχα να συμβάλλει στο πρόβλημα που εξετάζουμε καθώς περιπλέκει αρνητικά τον τρόπο με τον οποίο λειτουργεί ο εγκέφαλος και σχετίζεται άμεσα με την κατάθλιψη, άγχος και ADHD που αναφέρθηκαν παραπάνω [3], [8].

### 1.1.2 Επιπτώσεις στην Καθημερινή Ζωή

Το πρόβλημα της ανοργάνωτης ντουλάπας φέρει αρκετές επιπτώσεις στην καθημερινότητα ενός ανθρώπου, πολλές από τις οποίες μπορεί κανείς να αγνοήσει ή να μην τους δώσει ιδιαίτερη σημασία. Αυτό έχει ως αποτέλεσμα το πρόβλημα να εντείνεται και οι επιπτώσεις του από αόρατες, σταδιακά να γίνονται πιο εμφανείς και συνεπώς πολλές άλλες πτυχές της καθημερινής ζωής να μεταβάλλονται αρνητικά.

Η ακαταστασία μιας ντουλάπας με είδη ένδυσης επηρεάζει τον τρόπο που λειτουργεί ο εγκέφαλος αφού διαταράσσει την εστίαση και συγκέντρωση, προκαλώντας αρνητικά συναισθήματα όπως άγχος, πίεση και ντροπή, καθώς η οργάνωσή της προστίθεται στη λίστα με δουλειές προς διεκπεραίωση. Αυτά τα συναισθήματα ενισχύονται πολλές φορές μέσα στην ημέρα όταν ο άνθρωπος αναγκάζεται να ανοίξει την ντουλάπα του για να επιλέξει την ενδυμασία του. Μια ανοργάνωτη ντουλάπα δίνει ένα κακό ξεκίνημα σε μια παραγωγική

ημέρα κατά την οποία πρέπει τα συναισθήματα να μην επηρεάζουν την απόδοση. Παράλληλα, σπαταλιέται πάρα πολύς χρόνος αναζητώντας συγκεκριμένα αντικείμενα ένδυσης όταν επικρατεί ακαταστασία, που οδηγεί τον άνθρωπο στο να τα παρατάει και να φοράει τα ίδια σύνολα λόγω εύκολης πρόσβασης [9].

Επιπλέον, όταν δεν υπάρχει αρκετός χώρος λόγω ακαταστασίας, ανεξάρτητα από το μέγεθος της ντουλάπας, αποτρέπεται η αγορά νέων ειδών ένδυσης ακόμα και αν ο άνθρωπος τα χρειάζεται ή τα θέλει πολύ ώστε να μην καταλήξουν και αυτά στο χάος. Ωστόσο, μπορεί να προκαλέσει και περιττά έξοδα καθώς μια υπερπλήρης ντουλάπα, πολλές φορές αναγκάζει τον άνθρωπο να αγοράσει καινούρια ρούχα προκειμένου να αντικαταστήσει αυτά που έχουν χαθεί ή έχει να φορέσει πολύ καιρό σε σημείο που τα έχει ξεχάσει [9].

### 1.1.3 Ανάγκη για Ψηφιακή Λύση

Στη σημερινή εποχή οι άνθρωποι αναζητούν λύσεις στα προβλήματά τους μέσω διαδικτύου, είτε αυτά είναι μικρά όπως η αφαίρεση ενός λεκέ από ένα ρούχο είτε πιο σύνθετα όπως ψυχολογικά προβλήματα ή υγείας. Αυτές οι λύσεις μπορεί να έχουν τη μορφή ενός άρθρου, μιας ανάρτησης σε ένα blog ή μιας ιστοσελίδας/διαδικτυακής εφαρμογής [2].

Οι άνθρωποι επιλέγουν το διαδίκτυο για την αναζήτηση λύσεων στα προβλήματά τους διότι οι απαντήσεις είναι πιο άμεσες και ξεκάθαρες στην εύρεση οποιαδήποτε στιγμή και οπουδήποτε, εύκολα προσβάσιμες από διάφορες συσκευές και ποικίλες χάρη στις άφθονες πηγές που υπάρχουν με ενημερωμένες πληροφορίες. Γι' αυτούς τους λόγους τα τελευταία χρόνια έχει μειωθεί κατά πολύ η χρήση βιβλίων ή εγκυκλοπαιδειών για την αναζήτηση λύσεων και πληροφοριών [2].

Η διαδικτυακή εφαρμογή του “Digital Wardrobe” έρχεται να δώσει την ψηφιακή λύση στο πρόβλημα καθώς μπορεί να χρησιμοποιηθεί από κάθε άνθρωπο ανεξαρτήτως του επιπέδου εξοικείωσης χρήστης με το διαδίκτυο χάρη στην απλότητα της διεπαφής χρήστη (UI). Επίσης, μπορεί να το επισκεφτεί από οποιαδήποτε συσκευή και οποιαδήποτε χρονική στιγμή και τοποθεσία. Έτσι, η λύση στην ακαταστασία μιας ντουλάπας γίνεται άμεσα προσβάσιμη χάρη στη διαδικτυακή εφαρμογή μας, παρέχοντας τη γρηγορότερη και αποδοτικότερη εξυπηρέτηση στους χρήστες. Παράλληλα αυξάνει την παραγωγικότητα και το οργανωτικό πνεύμα, βελτιώνει την καθημερινότητα και μειώνει το άγχος που δημιουργείται κατά την επιλογή συνόλων και τον χρόνο που σπαταλιέται σε αυτό. Τέλος, χάρη στα υπόλοιπα χαρακτηριστικά λύνονται και άλλα προβλήματα που μπορεί να μην αναγνώριζαν στη συμπεριφορά τους οι χρήστες, όπως αυτό των “Travel Lists” που φροντίζει οι χρήστες να

δημιουργούν λίστες με ρούχα που θέλουν να πάρουν μαζί τους σε κάποιο ταξίδι ώστε να τους απαλλάξει από το πρόβλημα του να ξεχάσουν κάποιο από αυτά [10].

Αξίζει να σημειωθεί ότι υπάρχει μια παρόμοια εφαρμογή, γνωστή ως Whering, η οποία προσφέρει παρόμοιες λειτουργίες σχετικές με την οργάνωση και την ψηφιακή διαχείριση ντουλάπας. Ωστόσο, η εφαρμογή μας διαφοροποιείται προσφέροντας κάποιες επιπλέον λειτουργίες που δεν περιλαμβάνονται στο Whering, όπως οι ταξιδιωτικές λίστες και η δημιουργία περιστάσεων. Παράλληλα, το Whering διαθέτει κάποιες δυνατότητες που δεν έχουν υλοποιηθεί στην παρούσα εφαρμογή. Είναι σημαντικό να τονιστεί ότι η ανάπτυξη του Digital Wardrobe δεν επηρεάστηκε ή εμπνεύστηκε από τη συγκεκριμένη εφαρμογή. Οι δύο εφαρμογές συνυπάρχουν στον ίδιο χώρο και εξυπηρετούν παρόμοιες ανάγκες των χρηστών με διαφορετικές προσεγγίσεις και χαρακτηριστικά.

## 1.2 Σκοπός της Εφαρμογής

Ο σκοπός της διαδικτυακής εφαρμογής “Digital Wardrobe” βασίζεται στο πρόβλημα που αναφέρθηκε παραπάνω με την ανοργάνωση και την ακαταστασία μιας ντουλάπας με είδη ένδυσης. Μέσω των δυνατοτήτων που προσφέρει η εφαρμογή, δίνεται έμφαση στην οργάνωση των ρούχων σε μια ψηφιακή ντουλάπα κάτι που μετέπειτα μπορεί να δώσει το κίνητρο στους χρήστες να είναι περισσότερο οργανωμένοι και στην πραγματική ζωή. Παράλληλα, δίνεται έμφαση και στην αποφυγή σπατάλης χρόνου. Παρακάτω θα αναλύσουμε όλους τους σκοπούς του “Digital Wardrobe” αλλά και τις πτυχές της καθημερινότητας που αυτή η εφαρμογή θα βελτιώσει κατά τη χρήση της.

### 1.2.1 Βασικός Σκοπός

Ο βασικός σκοπός της διαδικτυακής εφαρμογής “Digital Wardrobe” είναι η παροχή μιας απλής διαδικασίας δημιουργίας και οργάνωσης της ψηφιακής γκαρνταρόμπας των χρηστών. Μέσω της εφαρμογής, οι χρήστες προσθέτουν ρούχα στην ντουλάπα τους και κατά την προσθήκη μπορούν να τα κατηγοριοποιήσουν με βάσει το είδος, χρώμα, εποχή κ.α., πράξη που μπορεί να τους ωθήσει στο να οργανώσουν και την πραγματική τους ντουλάπα με τον ίδιο τρόπο. Παράλληλα, η εξοικονόμηση χρόνου αποτελεί ακόμη έναν από τους πιο βασικούς μας στόχους καθώς θέλουμε να δώσουμε ένα ευχάριστο ξεκίνημα στην ημέρα των χρηστών, δίνοντας τους τη δυνατότητα να δημιουργούν τα δικά τους σύνολα και να επιλέγουν γρήγορα το κατάλληλο για κάθε περίσταση χωρίς να σπαταλάνε χρόνο σκεπτόμενοι τι θα φορέσουν.

Όπως αναφέρθηκε παραπάνω, η δημιουργία συνόλων είναι μια λειτουργία που στοχεύει στην εξοικονόμηση χρόνου. Την ίδια στιγμή όμως, αποσκοπεί στην αύξηση της δημιουργικότητας των χρηστών καθώς μέσω της λειτουργίας αυτής μπορούν να πειραματιστούν με τους συνδυασμούς των ρούχων τους, βρίσκοντας έτσι αυτό που τους ταιριάζει καλύτερα και αναπτύσσοντας το προσωπικό τους στυλ, κάτι που στο παρελθόν μπορεί να τους έλειπε. Χάρη σε ακόμη μια λειτουργία με όνομα “Travel Lists” (λίστες ταξιδιών) οι χρήστες μπορούν να δημιουργήσουν μια λίστα με τα ρούχα που επιθυμούν να πάρουν μαζί σε κάποιο ταξίδι. Ο σκοπός πίσω από τη λειτουργία αυτή είναι η διευκόλυνση των χρηστών αφού μπορούν να δημιουργήσουν και να επεξεργαστούν μια λίστα καιρό πριν το ταξίδι και όταν έρθει η ώρα να τους εξοικονομήσει σημαντικό χρόνο προετοιμασίας αλλά και να μειώσει την πιθανότητα να ξεχάσουν σημαντικά αντικείμενα.

Το “Digital Wardrobe” αποσκοπεί επίσης στην αποφυγή περιττών εξόδων. Η άμεση πρόσβαση στην ψηφιακή ντουλάπα επιτρέπει στους χρήστες να ελέγχουν τα υπάρχοντα ρούχα τους προτού προβούν σε αγορά καινούριων, διασφαλίζοντας ότι δεν κατέχει άλλα παρόμοια τους ή ότι αυτά δεν ταιριάζουν με άλλα είδη ένδυσης στην ντουλάπα τους ώστε να τα συνδυάσουν. Παράλληλα, προωθούνται πιο συνειδητές καταναλωτικές συνήθειες, βιοηθώντας τους χρήστες να καλύψουν τα κενά που εντοπίζουν στην γκαρνταρόμπα τους, ενθαρρύνοντας πιο υπεύθυνες αγορές και εξασφαλίζοντας έτσι τη βιωσιμότητα στην ντουλάπα τους.

### 1.2.2 Βελτίωση Καθημερινότητας

Με τη χρήση του “Digital Wardrobe” βελτιώνονται πολλές πτυχές της καθημερινότητας των ανθρώπων. Αρχικά, σε περίπτωση που κάποιοι μη εξοικειωμένοι χρήστες του διαδικτύου το χρησιμοποιούν, μπορούν να κάνουν τα πρώτα τους βήματα στην εξοικείωση καθώς η διαδικτυακή εφαρμογή ακολουθεί το μοτίβο μιας απλά δομημένης και σχεδιασμένης σελίδας έτσι ώστε να ανταποκρίνεται σε όλες τις ηλικιακές ομάδες. Η εξοικονόμηση χρόνου χάρη στη δημιουργία συνόλων αποτελεί σημαντικό πλεονέκτημα αφού μειώνει το άγχος και τη δυσκολία επιλογής ρούχων το πρωί, διευκολύνοντας την έναρξη της ημέρας με έναν πιο ευχάριστο και αποτελεσματικό τρόπο.

Μέσω της διαδικασίας δημιουργίας συνόλων ενισχύεται η δημιουργικότητα των χρηστών. Κάποιοι από αυτούς μπορεί να αισθάνονται ότι δεν έχουν ανακαλύψει ακόμη το στυλ που τους ταιριάζει, κατάσταση που συνήθως οδηγεί σε χαμηλή αυτοεκτίμηση λόγω της αβεβαιότητας όσον αφορά την καλύτερη δυνατή εμφάνιση. Όμως, χάρη στη λειτουργία αυτή

οι χρήστες μπορούν εύκολα να πειραματιστούν με τον συνδυασμό ρούχων, ανακαλύπτοντας αυτούς που μπορεί να μην είχαν σκεφτεί στο παρελθόν, πράγμα που τους κάνει να αισθάνονται αυτοπεποίθηση για τον εαυτό τους και τις επιλογές τους. Η δυνατότητα οπτικής αναπαράστασής τους ενισχύει την οργανωτική ικανότητα και την ψυχολογία, κινητροδοτεί και γεννά τη δημιουργική σκέψη στους χρήστες ώστε να οργανώσουν την ακατάστατη ντουλάπα τους, δίνοντας έτσι αξία σε αυτή και καθιστώντας την έναν ευχάριστο καθημερινό προορισμό.

Η λειτουργία “Travel Lists” επιτρέπει στους χρήστες να προετοιμάζουν τις λίστες με τα ρούχα που θέλουν να πάρουν μαζί τους σε κάποιο ταξίδι. Με τη δυνατότητα δημιουργίας και διαχείρισης αυτών των λιστών, οι χρήστες οργανώνονται πιο αποτελεσματικά αφού απομακρύνει το στρες της επιλογής ρούχων την τελευταία στιγμή, αποτρέπεται το ενδεχόμενο να ξεχάσουν κάτι σημαντικό πίσω και αποφεύγεται η μεταφορά περιττών ειδών ένδυσης, οδηγώντας σε πιο ελαφριές αποσκευές. Επομένως, αυτή η λειτουργία προσφέρει ευελιξία και άμεση εξυπηρέτηση όπου και αν βρίσκονται οι χρήστες.

Τέλος, η δυνατότητα επισκόπησης των ειδών ένδυσης επιτρέπει στους χρήστες να τα επισκέπτονται οποτεδήποτε, αναζητώντας με βάσει οποιοδήποτε χαρακτηριστικό έτσι ώστε να αξιολογήσουν εάν υπάρχουν τυχόν ελλείψεις ή παρόμοια είδη στην γκαρνταρόμπα τους προτού προβούν σε κάποια περιττή αγορά, κάτι που ενθαρρύνει πιο ορθές καταναλωτικές συνήθειες. Παράλληλα, η δυνατότητα επισκόπησης μπορεί να τους οδηγήσει στο να χαρίσουν ή να δωρίσουν κάποια από αυτά σε ανθρώπους που τα έχουν ανάγκη, εφόσον δεν τα χρειάζονται πλέον ή θέλουν να τα αντικαταστήσουν με καινούρια.

## 1.3 Στόχοι και Προσδοκίες

Η διαδικτυακή εφαρμογή “Digital Wardrobe” σχεδιάστηκε με συγκεκριμένους στόχους που επιδιώκουν να βελτιώσουν πολλές πτυχές της ζωής των χρηστών της, ενώ παράλληλα διαμορφώνει προσδοκίες για θετικές αλλαγές σε αυτή. Παρακάτω θα αναλύσουμε αρχικά τους στόχους της διαδικτυακής εφαρμογής που θέλουμε να επιτύχουμε, δηλαδή τα μετρήσιμα αποτελέσματα, καθώς και τις προσδοκίες, με άλλα λόγια τα οφέλη που αναμένουμε να προκύψουν από τη χρήση της.

### 1.3.1 Στόχοι

Ένας βασικός στόχος της εφαρμογής μας είναι να βοηθήσουμε τους χρήστες να οργανώσουν, βάσει διαφόρων κατηγοριών, τα ρούχα τους σε ψηφιακή μορφή και έπειτα να τους ενθαρρύνουμε να κάνουν το ίδιο και στη φυσική ντουλάπα εφόσον αυτή είναι ακατάστατη. Την ίδια στιγμή στοχεύουμε στη μέγιστη δυνατή εξοικονόμηση χρόνου, κάτι που βασίζουμε στη λειτουργία δημιουργίας συνόλων ώστε οι χρήστες, σε περίπτωση που πιέζονται χρονικά, να έχουν διαθέσιμες επιλογές από τις οποίες θα διαλέγουν αυτή που ταιριάζει περισσότερο ανά περίσταση.

Επιπλέον, η ενίσχυση της δημιουργικότητας αλλά και της αυτοπεποίθησης των χρηστών αποτελεί ακόμη έναν πολύ σημαντικό στόχο μας. Η λειτουργία δημιουργίας συνόλων δίνει την ευκαιρία στους χρήστες να πειραματιστούν με τον συνδυασμό των ειδών ένδυσης τους χωρίς να απαιτείται να τα δοκιμάσουν στην πραγματικότητα, αρκεί που θα εμφανίζονται σε εικονίδια το ένα κάτω από το άλλο στην προβολή συνόλου. Κάποιοι από αυτούς τους χρήστες που μπορεί να μην έχουν ακόμη ανακαλύψει το προσωπικό στυλ ντυσίματος τους, θα βρουν αυτή τη λειτουργία πολύ χρήσιμη καθώς ο πειραματισμός, η ανακάλυψη και η απόρριψη συνόλων θα έχει ως αποτέλεσμα να αισθάνονται αυτοπεποίθηση και σιγουριά για τις επιλογές τους.

Η σωστή οργάνωση και προετοιμασία μιας λίστας με ρούχα για κάποιο επικείμενο ταξίδι είναι ένας εξίσου σημαντικός στόχος για την εφαρμογή μας. Μέσω της λειτουργίας “Travel Lists” θέλουμε οι χρήστες να μπορούν να δημιουργήσουν μια λίστα όσο το δυνατόν νωρίτερα από την ημερομηνία αναχώρησης του ταξιδιού τους έτσι ώστε να αποφευχθεί οποιοδήποτε άγχος ή η πιθανότητα να ξεχάσουν κάτι σημαντικό πίσω. Τέλος, στοχεύουμε στη μείωση των περιττών εξόδων των χρηστών, βοηθώντας τους, προτού πραγματοποιήσουν μια αγορά, να βλέπουν τα ρούχα που ήδη κατέχουν προκειμένου να μην αγοράσουν άλλα παρόμοια ή που δεν ταιριάζουν με τα υπόλοιπα. Παράλληλα όμως θέλουμε μέσω αυτού να εντοπίζουν τις ελλείψεις στην γκαρνταρόμπα τους ή ρούχα που πλέον δεν είναι παράταιρα του στυλ τους και μπορούν να τα δωρίσουν.

### 1.3.2 Προσδοκίες

Με τη χρήση της εφαρμογής “Digital Wardrobe” αναμένεται να βελτιωθεί η ψυχολογία των χρηστών έπειτα από τη μείωση του άγχους που προκύπτει από την ανοργάνωτη ντουλάπα και τη δυσκολία επιλογής ρούχων για καθημερινές δραστηριότητες. Παράλληλα, αναμένεται οι χρήστες να αισθάνονται λιγότερη πίεση χρόνου από την έναρξη της ημέρας χάρη στο χρόνο που τους εξοικονομούν τα ήδη έτοιμα σύνολα.

Η τακτική χρήση της διαδικτυακής εφαρμογής για την οργάνωση των ειδών ένδυσης προσδοκάται να αναπτύξει την αίσθηση ελέγχου της γκαρνταρόμπας και η λειτουργία δημιουργίας συνόλων να ενισχύσει ή να καλλιεργήσει την αυτοπεποίθησή για το προσωπικό τους στυλ. Επιπλέον, με τη χρήση της λειτουργίας “Travel Lists” προσδοκούμε οι χρήστες να νιώθουν μεγαλύτερη ανυπομονησία και σιγουριά για τα ταξίδια τους καθώς θα έχουν ήδη έτοιμες τις λίστες τους κάτι που θα τους απαλλάξει από το στρες της προετοιμασίας.

Τέλος, η χρήση της αναμένεται να συμβάλει σημαντικά στην εξοικονόμηση χρημάτων των χρηστών, αφού τους παρέχει τη δυνατότητα να προβάλλουν τα ρούχα που ήδη κατέχουν πριν προχωρήσουν σε νέες αγορές. Αυτό μειώνει τα περιττά έξοδα, αποτρέποντας την αγορά παρόμοιων ή μη απαραίτητων ειδών. Ωστόσο, στην περίπτωση που ορισμένοι χρήστες θέλουν να αγοράσουν ένα νέο ρούχο για να αντικαταστήσουν κάποιο που δεν χρησιμοποιούν πλέον, προσδοκούμε ότι θα ενθαρρυνθούν να το δωρίσουν ή να το ανακυκλώσουν, κάτι από το οποίο θα επωφεληθούν τόσο οι χρήστες με μια βιώσιμη ντουλάπα όσο και οι κοινωνικές ομάδες που έχουν ανάγκη. Έτσι, αναμένεται η εφαρμογή να συμβάλλει στη μείωση του περιβαλλοντικού αποτυπώματος μέσω υπεύθυνων αγορών και ορθής διαχείρισης ρούχων.

## 1.4 Δομή της εφαρμογής

Η διαδικτυακή εφαρμογή “Digital Wardrobe” σχεδιάστηκε με κύρια έμφαση στην απλότητα του σχεδιασμού της και στην ευχρηστία, έτσι ώστε να είναι προσιτή σε όλους τους χρήστες ανεξαρτήτως ηλικίας ή επιπέδου εξοικείωσης με το διαδίκτυο και τις ιστοσελίδες. Ένας ακόμη λόγος που η εφαρμογή σχεδιάστηκε με τέτοιο τρόπο ώστε να είναι απλή, είναι διότι θέλουμε να δώσουμε την ευκαιρία στους άπειρους χρήστες να αποκτήσουν την εξοικείωση με το διαδίκτυο μέσω του “Digital Wardrobe”, ενώ επίσης θέλουμε οι καθημερινοί χρήστες αυτού να αισθάνονται άνετα με τη χρήση της εφαρμογής χωρίς να συναντούν εμπόδια και πολυπλοκότητες στις λειτουργίες [11].

Για την ανάπτυξη της εφαρμογής τόσο στο Back-End όσο και στο Front-End, χρησιμοποιήθηκαν σύγχρονα εργαλεία και τεχνολογίες έτσι ώστε να εξασφαλίσουμε την ταχύτητα, την ασφάλεια και την ευχρηστία της. Στο Back-End δόθηκε έμφαση στην ξεκάθαρη διαχείριση και την ασφάλεια δεδομένων των χρηστών, ενώ στο Front-End, εστιάσαμε στη δημιουργία μιας φιλικής προς τους χρήστες διεπαφής. Με αυτή τη δομή, επιτυγχάνεται η ισορροπία μεταξύ απλότητας και αποτελεσματικότητας, παρέχοντας μια ομαλή εμπειρία στους χρήστες [11].

### 1.4.1 Back-End

Για την ανάπτυξη του Back-End εστιάσαμε στην καλύτερη δυνατή διαχείριση των δεδομένων των χρηστών και την ασφάλειά τους. Συνοπτικά, τα κύρια εργαλεία που συνέβαλαν στην ανάπτυξή του ήταν η γλώσσα προγραμματισμού JavaScript (JS) που κυρίως χρησιμοποιείται για τη δημιουργία διαδραστικών web εφαρμογών διεπαφών χρήστη [12], το Node.js το οποίο είναι ένα περιβάλλον εκτέλεσης JavaScript που επιτρέπει τη χρήση της για τη δημιουργία server-side εφαρμογών και το Express.js, framework του προαναφερθέντος, που απλοποιεί τη δημιουργία και διαχείριση servers και routes [13].

Παράλληλα, για τη βάση δεδομένων χρησιμοποιήθηκε το MongoDB που αποθηκεύει δεδομένα σε μορφή εγγράφων JSON και όχι σε πίνακες όπως κάνουν οι περισσότερες βάσεις. Χρησιμοποιεί μια ευέλικτη δομή για την αποθήκευση των δεδομένων, κάτι που επιτρέπει τη διαχείριση δομημένων ή μη-δομημένων μορφών τους [14]. Το Mongoose, μια βιβλιοθήκη για Node.js, λειτουργεί ως διαμεσολαβητής παρέχοντας ένα schema-based μοντέλο για την οργάνωση, επικύρωση και χειρισμό των δεδομένων, διευκολύνοντας έτσι την αλληλεπίδραση μεταξύ της εφαρμογής και του MongoDB [15].

### 1.4.2 Front-End

Για την ανάπτυξη του Front-End εστιάσαμε στη δημιουργία μιας φιλικής, εύκολης και απλής διεπαφής για τους χρήστες της διαδικτυακής εφαρμογής. Δημιουργήθηκε χρησιμοποιώντας το React.js που είναι μια βιβλιοθήκη ανάπτυξης UI βασισμένη στη JavaScript [16], σε συνδυασμό με τις γλώσσες προγραμματισμού JavaScript [12], HTML [17] και CSS [18], που όλες μαζί πραγματοποιούν την αλληλεπίδραση με τους χρήστες. Αυτές οι τεχνολογίες επιτρέπουν τη δημιουργία μιας δυναμικής και διαδραστικής διεπαφής χρήστη, διασφαλίζοντας βελτιστοποίηση ταχύτητας, απλότητας και ευχρηστίας [19].

Όλες οι λειτουργίες είναι σαφώς οργανωμένες με εύκολα προσβάσιμα μενού από μια επικεφαλίδα και ενότητες, ώστε οι χρήστες να μπορούν να ολοκληρώνουν κάθε ενέργεια τους χωρίς περιττές πολυπλοκότητες και εμπόδια. Τέλος, το React.js συμβάλλει στην ταχύτερη απόδοση των σελίδων, προσφέροντας μια ομαλή εμπειρία χρήστη χωρίς καθυστερήσεις, ακόμη και σε περιβάλλοντα με περιορισμένους πόρους [16].



## 2 BACK-END

Στο κεφάλαιο αυτό, θα αναλύσουμε τη δομή και τη λειτουργία του Back-End της διαδικτυακής εφαρμογής “Digital Wardrobe”. Θα εξετάσουμε τα κύρια εργαλεία, τις τεχνολογίες και τις βιβλιοθήκες που χρησιμοποιήθηκαν, τους λόγους πίσω από τις συγκεκριμένες επιλογές και πώς συμβάλλουν συνολικά στην ομαλή και σταθερή λειτουργία της εφαρμογής. Επιπλέον, θα παρουσιαστούν στιγμιότυπα από τα σημαντικότερα τμήματα του κώδικα και θα εστιάσουμε στους τρόπους με τους οποίους η εφαρμογή εξασφαλίζει την ασφάλεια και την προστασία των δεδομένων των χρηστών. Τέλος, θα γίνει μια αναδρομή στα εμπόδια και τις προκλήσεις που συναντήσαμε κατά την ανάπτυξη και στο πως αντιμετωπίστηκαν αυτά.

### 2.1 Εισαγωγή στο Back-End

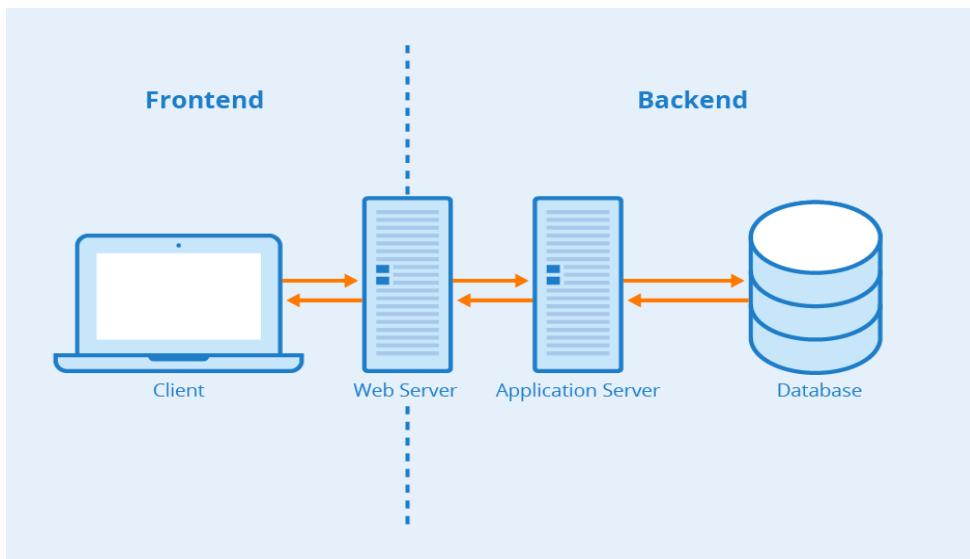
Το Back-End της εφαρμογής διαχειρίζεται τις λειτουργίες που αφορούν την αποθήκευση και διαχείριση δεδομένων, καθώς και τη σύνδεση με τη βάση δεδομένων. Ο κύριος σκοπός του είναι να προσφέρει την ομαλή και σταθερή αλληλεπίδραση μεταξύ των χρηστών και των λειτουργιών που διαθέτει η εφαρμογή [20]. Συνοπτικά, αυτές τις λειτουργίες αποτελούν η προσθήκη/δημιουργία, επεξεργασία/ενημέρωση και διαγραφή προφίλ χρήστη, ρούχων, συνόλων, περιστάσεων και ταξιδιωτικών λιστών. Είναι υπεύθυνο για την ασφαλή αποθήκευση των πληροφοριών που εισάγουν οι χρήστες και τη διαχείριση όλων των αιτημάτων που προέρχονται από το Front-End.

#### 2.1.1 Ρόλος στην εφαρμογή

Ο κεντρικός ρόλος του Back-End της εφαρμογής είναι να διαχειρίζεται και να επεξεργάζεται τα αιτήματα που λαμβάνει από το Front-End. Αυτά τα αιτήματα μπορεί, για παράδειγμα, να είναι η διαγραφή του προφίλ του χρήστη, η επεξεργασία ενός ρούχου και η δημιουργία ενός συνόλου ειδών ένδυσης. Έπειτα από τη λήψη αυτών των αιτημάτων, το Back-End διαχειρίζεται τα δεδομένα των χρηστών, τα οποία στη συνέχεια με ασφάλεια αποθηκεύονται στη βάση δεδομένων μέσω της βιβλιοθήκης Mongoose [15], [21]. Παράλληλα, εξασφαλίζει την προστασία των δεδομένων που εισάγονται με τη χρήση μηχανισμών ασφαλείας και απαντάει στα αιτήματα των χρηστών, ενημερώνοντας το Front-End για τις απαραίτητες ενέργειες.

### 2.1.2 Επικοινωνία με το Front-End

Στη διαδικτυακή μας εφαρμογή, το Back-End και το Front-End επικοινωνούν μέσω HTTP αιτημάτων. Το Front-End στέλνει αιτήματα (GET, POST, PUT, DELETE) στο Back-End, τα οποία διαχειρίζονται από τις διαδρομές (routes) που δημιουργήθηκαν για κάθε λειτουργία (clothes\_routes.js, outfits\_routes.js κλπ.). Τα αιτήματα τα λαμβάνει το Back-End για την επεξεργασία δεδομένων και στη συνέχεια επικοινωνεί με τη βάση δεδομένων MongoDB μέσω του Mongoose [15], [21] ώστε να πραγματοποιηθεί η αποθήκευση ή η ανάκτηση των απαραίτητων δεδομένων. Με την ολοκλήρωση των παραπάνω, το Back-End επιστρέφει τις απαραίτητες πληροφορίες στο Front-End, το οποίο με τη σειρά του ενημερώνει τη διεπαφή χρήστη (UI) [22].



*Εικόνα 1: Η επικοινωνία του Front-End, Back-End και μιας βάσης δεδομένων.*

### 2.2 Αρχιτεκτονική και Σχεδιασμός

Η αρχιτεκτονική του Back-End της εφαρμογής “Digital Wardrobe” σχεδιάστηκε με στόχους την απλότητα, την επεκτασιμότητα και την ασφάλεια. Σκοπός μας ήταν να αναπτύξουμε μια δομή η οποία θα ήταν τόσο ευέλικτη όσο και δυναμική που θα διαχειρίζεται τα δεδομένα με ασφάλεια και ταχύτητα. Η υλοποίηση βασίστηκε σε εργαλεία όπως το Node.js και το Express.js, που ήταν και τα κυρίαρχα στη διαχείριση αιτημάτων και τον έλεγχο ροής δεδομένων μεταξύ των χρηστών και της βάσης δεδομένων MongoDB [13], [14], [23].

### 2.2.1 Λόγος Σχεδιασμού

Η αρχιτεκτονική του Back-End της εφαρμογής βασίστηκε στην ιδέα ενός ευέλικτου και αποδοτικού συστήματος που μπορεί να διαχειριστεί τα αιτήματα, που προέρχονται από ενέργειες των χρηστών στο Front-End, με χαμηλή καθυστέρηση και υψηλή απόδοση [23]. Η επιλογή του Node.js προσφέρει ταχύτητα επεξεργασίας δεδομένων και αιτημάτων και γρήγορες αλληλεπιδράσεις μεταξύ χρηστών και διεπαφής ακόμα και σε περιπτώσεις μεγάλου φόρτου [24]. Το Express.js επιλέχθηκε για τη διαχείριση των διαδρομών (routes) και των αιτημάτων HTTP (GET, POST, PUT, DELETE), προσφέροντας μια απλή και αποδοτική υποδομή για την αλληλεπίδραση με το Front-End [25]. Η επιλογή των προαναφερθέντων, καθώς και άλλων εργαλείων, θα αναλυθούν στην επόμενη ενότητα [13].

### 2.2.2 Διαχείριση Δεδομένων και Απόκρισης

Η διαχείριση δεδομένων στην εφαρμογή γίνεται μέσω της βάσης δεδομένων MongoDB, η οποία τα έχει αποθηκευμένα σε μορφή JSON, κάτι που επιτρέπει ευέλικτη και επεκτάσιμη διαχείριση. Όλα τα αιτήματα που λαμβάνονται από το Express.js επεξεργάζονται κατάλληλα και ύστερα έρχεται σε επικοινωνία με τη MongoDB για την αποθήκευση και ανάκτηση δεδομένων. Τα αποτελέσματα επιστρέφονται στο Font-End σε μορφή JSON [14], [26]. Με τη χρήση της βιβλιοθήκης Mongoose, που προσφέρει ένα schema-based σύστημα, επιτυγχάνεται η οργάνωση και επικύρωση των δεδομένων [15], [27].

## 2.3 Τεχνολογίες και Εργαλεία

Η ανάπτυξη του Back-End της εφαρμογής βασίστηκε στη χρήση σύγχρονων τεχνολογιών και εργαλείων που προσφέρουν αποδοτικότητα, επεκτασιμότητα και ασφάλεια. Κάθε ένα από αυτά επιλέχθηκε με κύριους γνώμονες την καλύτερη δυνατή διαχείριση δεδομένων και την αποτελεσματική αλληλεπίδραση με το Front-End [28]. Σε αυτή την ενότητα, θα αναλυθούν οι γλώσσες προγραμματισμού, βάση δεδομένων, frameworks και περιβάλλοντα εκτέλεσης που χρησιμοποιήθηκαν, καθώς και οι βιβλιοθήκες που συμβάλλουν στην πλήρη και ομαλή λειτουργία του Back-End ώστε να επιτυγχάνονται με τον καλύτερο δυνατό τρόπο οι στόχοι μας [23]. Συγκεκριμένα θα γίνει αναφορά στην ευρεία χρήση τους, αλλά θα εστιάσουμε στους τρόπους με τους οποίους αξιοποιήθηκαν στη δόμηση του Back-End της εφαρμογής μας.

### 2.3.1 JavaScript

Η JavaScript είναι μια από τις πιο δημοφιλείς γλώσσες προγραμματισμού, που χρησιμοποιείται κυρίως για την ανάπτυξη διαδραστικών στοιχείων σε μια ιστοσελίδα [12]. Η προσαρμοστικότητα και η δυναμικότητά της είναι κάποια από τα μεγαλύτερα πλεονεκτήματά της καθώς επιτρέπουν στους χρήστες να εκτελεστεί απευθείας από τον περιηγητή ιστού, καθιστώντας την ιδανική για τη σχεδίαση Front-End εφαρμογών. Ταυτόχρονα μπορεί να χρησιμοποιηθεί και στο Back-End μέσω του Node.js, δίνοντας τη δυνατότητα ανάπτυξης μιας server-side λογικής με JavaScript [29].

Στην ανάπτυξη του Back-End της εφαρμογής μας, η JavaScript έχει κεντρικό ρόλο, υλοποιώντας όλες τις κύριες λειτουργίες. Χρησιμοποιείται μέσω του Node.js για τη δημιουργία του server και την επεξεργασία των HTTP αιτημάτων που προέρχονται από το Front-End [22]. Μέσω του Express.js, διαχειρίζεται τις διαδρομές (clothes\_routes.js, travellists\_routes.js κ.α.) για κάθε λειτουργία όπως η προσθήκη και επεξεργασία δεδομένων (π.χ. ρούχα, σύνολα, προφίλ χρήστη) [13]. Ταυτόχρονα, χρησιμοποιείται για τη σύνταξη των Mongoose Models που συνδέουν τη βάση δεδομένων MongoDB και τη διαχείριση της ασφάλειας μέσω jsonwebtoken για τη δημιουργία token αυθεντικοποίησης χρηστών [27], [30]. Συνοπτικά, διευθετεί τον server, τη ροή δεδομένων, τις διαδρομές, τη σύνδεση με τη βάση δεδομένων και την ασφάλεια στο Back-End [29].

### 2.3.2 Node.js

Το Node.js είναι ένα περιβάλλον εκτέλεσης JavaScript που επιτρέπει στους προγραμματιστές να τρέχουν JavaScript κώδικα στον διακομιστή, έξω από τον περιηγητή ιστού. Είναι εξαιρετικά αποδοτικό για τη δημιουργία real-time εφαρμογών, με αρχιτεκτονική event-driven, επιτρέποντας τη διαχείριση μεγάλου όγκου δεδομένων ταυτόχρονα χωρίς κάποια καθυστέρηση. Είναι δημοφιλές για την ανάπτυξη web servers και API. Η ποικιλία και η ευρεία υποστήριξη από βιβλιοθήκες όπως το Express.js, κάνουν το Node.js ιδανικό για την ανάπτυξη ταχύτατων και αποδοτικών εφαρμογών ιστού [23], [24], [31].

Στο Back-End της εφαρμογής μας, το Node.js χρησιμοποιείται για να λειτουργεί ως server-side πλαίσιο, ώστε να διαχειρίζεται όλες τις λειτουργίες και τα αιτήματα που προέρχονται από το Front-End. Τα αιτήματα τα επεξεργάζεται μέσω του Express.js ενώ επίσης χρησιμοποιείται για τη σύνδεση με τη βάση δεδομένων μέσω της βιβλιοθήκης Mongoose [25], [27]. Χάρη στην ασύγχρονη φύση του, εξασφαλίζει ότι η εφαρμογή μας

μπορεί να χειρίζεται πολλαπλές παράλληλες λειτουργίες χωρίς καθυστερήσεις, κάνοντάς την πιο αποδοτική. Χρησιμοποιείται επίσης για την επικοινωνία με εξωτερικές υπηρεσίες, όπως η διαχείριση αρχείων εικόνας που αποθηκεύονται στο server και η αυθεντικοποίηση χρηστών μέσω JWT tokens [30].

### 2.3.3 Express.js

Το Express.js είναι το πιο δημοφιλές web framework για το Node.js. Έχει σχεδιαστεί για τη δημιουργία εφαρμογών ιστού και APIs. Απλοποιεί τη διαδικασία ανάπτυξης server-side εφαρμογών, καθώς εξοικονομεί χρόνο από κουραστικές και χρονοβόρες διαδικασίες και δίνει τη δυνατότητα στους προγραμματιστές να εστιάσουν σε πιο σημαντικές εργασίες. Παρέχει ένα σύνολο εργαλείων και λειτουργιών για τη διαχείριση αιτημάτων HTTP (GET, POST, PUT, DELETE) [32], την ανάπτυξη APIs και τη διαχείριση των διαδρομών (routes). Το Express.js επιτρέπει στους προγραμματιστές να διαμορφώνουν εύκολα web servers με ελάχιστο κώδικα και μεγάλη ευελιξία [25], [33], [34].

Στην εφαρμογή μας, το Express.js χρησιμοποιείται για να διαχειρίζεται τα αιτήματα που αποστέλλονται από το Front-End στο Back-End. Με αυτό δημιουργήθηκαν οι διαδρομές (routes) για λειτουργίες όπως η προσθήκη/δημιουργία, επεξεργασία και διαγραφή διαφόρων αντικειμένων εντός εφαρμογής όπως προφίλ χρηστών, ρούχα και συνόλων. Χρησιμοποιεί διάφορα middleware, όπως για παράδειγμα για την επεξεργασία των δεδομένων που αποστέλλονται μέσω των φορμών και για την επαλήθευση ταυτότητας των χρηστών με JWT tokens, διασφαλίζοντας ότι μόνο οι εξουσιοδοτημένοι χρήστες έχουν πρόσβαση στα δεδομένα τους [30]. Χάρη στην απλότητα και ταχύτητα αυτού του framework, η αλληλεπίδραση μεταξύ των χρηστών και της βάσης δεδομένων διατηρείται ομαλή μέσω των διαδρομών που έχουν δημιουργηθεί.

### 2.3.4 Βάση Δεδομένων

Για την επιλογή της βάσης δεδομένων του “Digital Wardrobe” βασιστήκαμε στην ανάγκη για μια ευέλικτη και επεκτάσιμη βάση που να διαχειρίζεται τόσο δομημένα όσο και μη δομημένα δεδομένα, αποθηκεύοντας τα σε μορφή εγγράφων JSON. Έτσι καταλήξαμε στην επιλογή χρήσης της MongoDB, μια μη σχεσιακή (NoSQL) βάση δεδομένων που επιτελεί όλα τα παραπάνω. Είναι κατάλληλη για εφαρμογές που απαιτούν ταχύτητα και ευελιξία στη διαχείριση δεδομένων [14], [26]. Παρακάτω θα αναλυθεί η χρήση της MongoDB και του

Mongoose, που εξυπηρετεί ως μεσολαβητής προσφέροντας ένα schema-based σύστημα για την οργάνωση των δεδομένων, επικύρωση και επικοινωνία μεταξύ Back-End και βάσης [15], [21], [27].

### 2.3.4.1 MongoDB

Η MongoDB είναι μια δημοφιλής μη σχεσιακή (NoSQL) βάση δεδομένων που αποθηκεύει δομημένα και μη δομημένα δεδομένα σε μορφή εγγράφων JSON. Σε αντίθεση με τις σχεσιακές βάσεις (SQL) που βασίζονται σε πίνακες, η MongoDB προσφέρει μεγαλύτερη ευελιξία καθώς δεν απαιτεί σταθερά σχήματα για την αποθήκευση δεδομένων, αλλά συλλογές (collections) για να τα αποθηκεύσει, προσφέροντας δυνατότητες όπως ταχύτητα και ευκολία επέκτασης και διαχείρισης μεγάλου όγκου δεδομένων κάτι που την καθιστά κατάλληλη για real-time εφαρμογές αλλά και αυτές που απαιτούν γρήγορη πρόσβαση και ενημέρωση δεδομένων [14], [26].

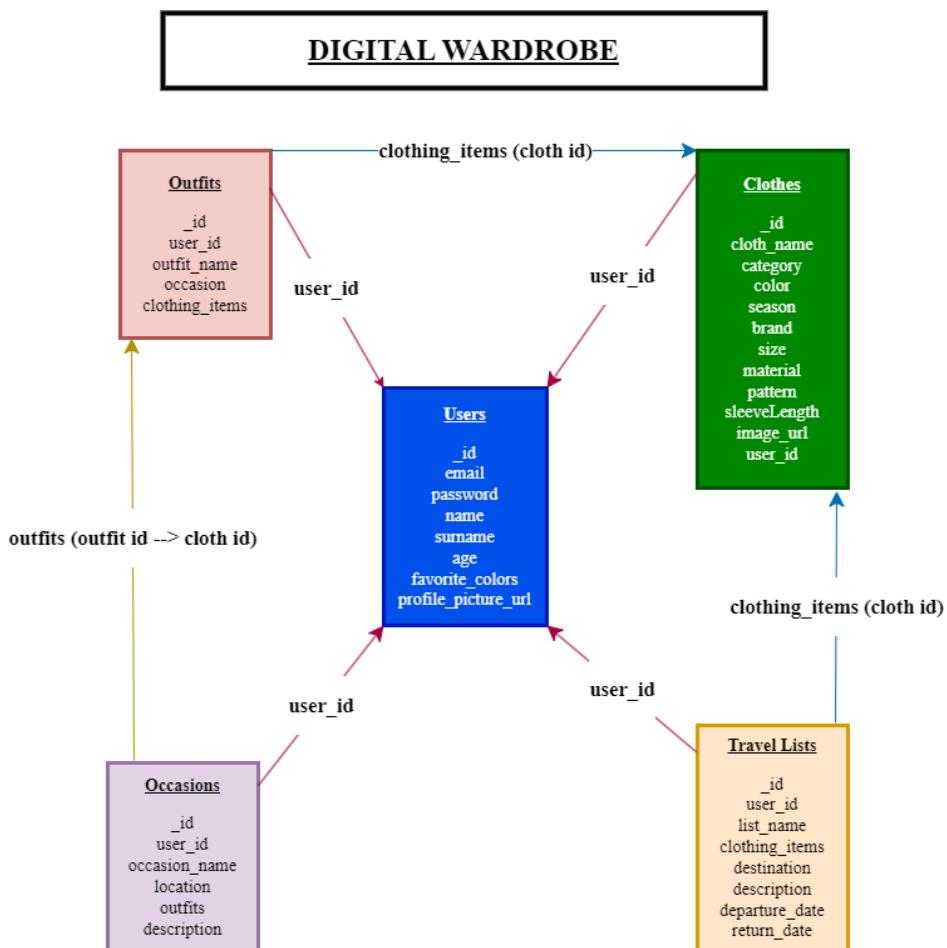
Στην εφαρμογή μας, η MongoDB αποθηκεύει τα δεδομένα σε συλλογές που αντιστοιχούν σε κάθε βασική λειτουργία που προσφέρουμε. Όπως φαίνεται και στην εικόνα 2, η βάση μας με όνομα “digwar” αποτελείται από πέντε κύριες συλλογές (collections): **Clothes**, **Occasions**, **Outfits**, **Travel Lists** και **Users**.

The screenshot shows the MongoDB Compass interface for a database named 'digwar'. The 'Collections' tab is selected. It displays five collections: Clothes, Occasions, Outfits, Travel Lists, and Users. Each collection has its storage size, number of documents, average document size, index count, and total index size listed.

Collection	Storage size:	Documents:	Avg. document size:	Indexes:	Total index size:
Clothes	20.48 kB	35	316.00 B	1	36.86 kB
Occasions	20.48 kB	9	218.00 B	1	36.86 kB
Outfits	20.48 kB	13	199.00 B	1	36.86 kB
Travel Lists	20.48 kB	5	276.00 B	1	36.86 kB
Users	20.48 kB	3	295.00 B	2	73.73 kB

**Εικόνα 2:** Οι συλλογές της βάσης δεδομένων του “Digital Wardrobe” όπως φαίνονται στην εφαρμογή MongoDB Compass.

Κάθε συλλογή περιέχει έγγραφα (documents) που αποθηκεύουν πληροφορίες για κάθε μια από αυτές σε πολλά διαφορετικά πεδία. Καθώς η εφαρμογή μας πρέπει να συνδέει τους χρήστες με τα αντικείμενα που δημιουργούν στην ιστοσελίδα, το αναγνωριστικό των χρηστών (**user\_id**) είναι παρόν σε κάθε έγγραφο όλων των συλλογών. Με τον ίδιο τρόπο, υπάρχουν και άλλα αναγνωριστικά που χρησιμοποιούνται με τον ίδιο σκοπό ώστε να συνδέονται μεταξύ τους, δημιουργώντας παραπάνω σχέσεις μεταξύ τους όπως φαίνεται και στην τρίτη εικόνα με το διάγραμμα σχήματος της βάσης δεδομένων [28], [35].



**Εικόνα 3:** Διάγραμμα σχήματος βάσης δεδομένων του “Digital Wardrobe” (δημιουργήθηκε στο draw.io).

### 2.3.4.2 Mongoose

Το Mongoose είναι μια δημοφιλής βιβλιοθήκη για το Node.js, της οποίας ο ρόλος είναι να δουλεύει με τη βάση δεδομένων MongoDB. Χάρη στο schema-based σύστημα που παρέχει,

επιτρέπει τη δημιουργία δομημένων δεδομένων, ακόμα και σε μια μη σχεσιακή (NoSQL) βάση όπως η MongoDB. Για την οργάνωση των δεδομένων χρησιμοποιούνται μοντέλα, προσφέροντας δυνατότητες όπως η επικύρωση δεδομένων, hooks για παρακολούθηση αλλαγών και middleware που διευκολύνουν την επικοινωνία μεταξύ της εφαρμογής και της MongoDB [15], [27].

Στην εφαρμογή μας, το Mongoose χρησιμοποιείται ως εργαλείο για την επικοινωνία με τη βάση δεδομένων MongoDB και για τη δημιουργία μοντέλων που ορίζουν τη δομή των συλλογών (collections). Συγκεκριμένα, η σύνδεση της εφαρμογής με τη βάση δεδομένων πραγματοποιείται μέσω της συνάρτησης “**mongoose.connect()**” εντός του κεντρικού αρχείου **app.js** όπως παρατηρείται στην παρακάτω εικόνα [21].

```
20  mongoose.connect(mongoURI, {
21    useNewUrlParser: true,
22    useUnifiedTopology: true,
23  }).then(() => {
24    console.log('Connected to MongoDB');
25  }).catch((error) => {
26    console.error('Error connecting to MongoDB', error);
27  });
```

**Εικόνα 4:** Τμήμα κώδικα στο αρχείο **app.js** που συνδέει το Back-End με τη βάση δεδομένων *MongoDB*.

Για τους σκοπούς της εφαρμογής δημιουργήσαμε μοντέλα για κάθε βασικό αντικείμενο-λειτουργία, δηλαδή, τα ρούχα (Clothes), τα σύνολα (Outfits), τις περιστάσεις (Occasions), τις λίστες ταξιδιών (Travel Lists) και τα προφίλ των χρηστών (Users). Κάθε ένα από αυτά τα μοντέλα περιέχουν σχήματα (schemas) που ορίζουν τις ιδιότητες και τους τύπους δεδομένων που επιτρέπεται να αποθηκευτούν, τα απαιτούμενα δεδομένα, ενώ ταυτόχρονα ενσωματώνουν επικυρώσεις για την ακεραιότητα των δεδομένων. Ορισμένα πεδία είναι πίνακες ώστε να μπορούν να αποθηκεύσουν πολλαπλές τιμές ή αντικείμενα [28].

Για παράδειγμα, το μοντέλο των ρούχων, περιλαμβάνει απαιτούμενα πεδία όπως το όνομα του ρούχου (**cloth\_name**), η κατηγορία (**category**), το αναγνωριστικό χρήστη (**user\_id**) κ.α. με τη συνοδεία της ετικέτας “**required: true**”. Επιπλέον, το μοντέλο περιλαμβάνει πεδία όπως η εικόνα (**image**) και άλλες πληροφορίες που αφορούν την περιγραφή και τις ιδιότητες του κάθε ρούχου. Ο κώδικας που περιγράφεται φαίνεται αναλυμένος στην πέμπτη εικόνα με όλες τις λεπτομέρειες των συμπεριλαμβανομένων πεδίων.

```
D: > Apps > Node.js > Digital_Wardrobe > models > js clothes.js > ...
1  const mongoose = require('mongoose');
2  const { Schema } = require('mongoose');
3
4  const clothSchema = new mongoose.Schema({
5    cloth_name: { type: String, required: [true, 'Please type a name of the cloth'] },
6    category: { type: [String], required: [true, 'Please select one or more categories'] },
7    color: { type: [String], required: [true, 'Please type one or more colors'] },
8    season: { type: [String], required: [true, 'Please select one or more seasons'] },
9    brand: { type: String, default: '' },
10   size: { type: String, default: '' },
11   material: { type: String, default: '' },
12   pattern: { type: String, default: '' },
13   sleeveLength: { type: String, default: '' },
14   image_url: { type: String, required: true },
15   user_id: { type: Schema.Types.ObjectId, required: true, ref: 'User' }
16 }, { collection: 'Clothes'});
17
18 const Cloth = mongoose.model('Cloth', clothSchema);
19 module.exports = Cloth;
```

**Εικόνα 5:** Κόδικας των μοντέλου των ρούχων με όλα τα πεδία του (Στιγμιότυπο από το αρχείο *clothes.js* στην εφαρμογή *Visual Studio Code*).

### 2.3.5 Άλλες Βιβλιοθήκες

Εκτός από τα εργαλεία, τις τεχνολογίες και τις βιβλιοθήκες που αναλύθηκαν παραπάνω, για την εξυπηρέτηση των αναγκών της εφαρμογής μας χρησιμοποιήθηκαν αρκετές άλλες βιβλιοθήκες του Node.js ή μέσω του Express.js [25]. Κάποιες από αυτές τις βιβλιοθήκες αναλαμβάνουν τη διαχείριση αρχείων, την επικύρωση δεδομένων, την ασφάλεια και την αυθεντικοποίηση των χρηστών. Η χρήση τους στην ανάπτυξη του Back-End, επιτρέπει τη δημιουργία ενός αποδοτικού και ασφαλούς περιβάλλοντος για τους χρήστες, εξασφαλίζοντας παράλληλα την ομαλή και ορθή λειτουργία της εφαρμογής. Στις υπο-ενότητες που ακολουθούν θα σημειωθεί η χρησιμότητα αυτών των βιβλιοθηκών αλλά και η συμβολή τους στην ανάπτυξη και διαχείριση του Back-End.

#### 2.3.5.1 Multer

To **Multer** είναι ένα middleware για το Node.js που χρησιμοποιείται για την αποδοχή και διαχείριση αρχείων που αποστέλλονται μέσω φόρμας στο Express.js. Ειδικεύεται στην αποθήκευση και μεταφόρτωση αρχείων στο διακομιστή και τη διαχείριση δεδομένων multipart/form-data, το οποίο επιτρέπει επιπλέον τον χειρισμό των ορίων μεγέθους των

αρχείων, τον ορισμό φίλτρων αρχείων και την αποτελεσματική οργάνωση των μεταφορτωμένων αρχείων [36].

Στην εφαρμογή μας, το **Multer** χρησιμοποιείται για την αποθήκευση εικόνων προφίλ και ρούχων των χρηστών στους αντίστοιχους φακέλους (uploads/pfp και uploads/clothes), με αυτό να αποτελεί τον κυριότερο λόγω επιλογής του καθώς στόχος μας είναι η οργάνωση. Περιλαμβάνεται στα αρχεία διαδρομών όπου πραγματοποιείται η διαχείριση της αποστολής εικόνων, δηλαδή στα users\_routes.js και clothes\_routes.js, ενώ επίσης και στο κεντρικό αρχείο app.js του Back-End [36].

```
// Setup multer for handling profile picture uploads
const storage = multer.diskStorage({
  destination: function(req, file, cb) {
    cb(null, 'uploads/pfp/');
  },
  filename: function(req, file, cb) {
    const uniqueSuffix = Date.now() + '-' + Math.round(Math.random() * 1E9);
    cb(null, file.fieldname + '-' + uniqueSuffix + '.' + file.originalname.split('.').pop());
  }
});
const upload = multer({ storage: storage });

// Setup multer for handling profile picture uploads
const storage = multer.diskStorage({
  destination: function(req, file, cb) {
    cb(null, 'uploads/clothes/');
  },
  filename: function(req, file, cb) {
    const uniqueSuffix = Date.now() + '-' + Math.round(Math.random() * 1E9);
    cb(null, file.fieldname + '-' + uniqueSuffix + '.' + file.originalname.split('.').pop());
  }
});
const upload = multer({ storage: storage });
```

**Εικόνα 6:** Παράδειγμα τμήματος κώδικα χρήσης του Multer στο αρχείο διαδρομών των χρηστών (users\_routes.js).

### 2.3.5.2 JWT (jsonwebtoken)

Το **jsonwebtoken** είναι μια βιβλιοθήκη που χρησιμοποιείται για τη δημιουργία και επαλήθευση JSON Web Tokens (JWT), επιτρέποντας την ασφαλή ανταλλαγή πληροφοριών μεταξύ διαφορετικών υπηρεσιών ή εφαρμογών. Αυτά τα tokens χρησιμοποιούνται συνήθως για έλεγχο ταυτότητας και εξουσιοδότηση, καθώς μπορούν να περιέχουν πληροφορίες που επαληθεύουν την ταυτότητα ενός χρήστη, καθώς και τις άδειες του [30], [37].

Στην εφαρμογή μας, το **jsonwebtoken** χρησιμοποιείται για την παραγωγή **access** και **refresh** tokens κατά την ασφαλή αυθεντικοποίηση των χρηστών, εξασφαλίζοντας την επαλήθευση πρόσβασης σε προστατευμένες διαδρομές. Με τη χρήση των συναρτήσεων generateAccessToken και generateRefreshToken στο αρχείο token.js, δημιουργούνται τα δύο tokens. Το **access token** χρησιμοποιείται για την επαλήθευση ταυτότητας σε κάθε αίτημα, ενώ το **refresh token** επιτρέπει την ανανέωσή του [30], [37]. Στο αρχείο authenticateToken.js, χρησιμοποιείται για την επαλήθευση των access tokens που

λαμβάνονται από τα αιτήματα χρηστών, διασφαλίζοντας ότι το αίτημα προέρχεται από εξουσιοδοτημένους χρήστες. Περιλαμβάνεται επίσης στο αρχείο auth\_routes.js.

```
30   router.post('/signin', async (req, res) => {
31     try {
32       const { email, password } = req.body;
33       const user = await User.findOne({ email });
34       if (!user) {
35         return res.status(401).send('Invalid email or password');
36       }
37       const isMatch = await bcrypt.compare(password, user.password);
38       if (!isMatch) {
39         return res.status(401).send('Invalid email or password');
40       }
41       const accessToken = generateAccessToken(user);
42       const refreshToken = generateRefreshToken(user);
43       res.json({
44         message: 'Successfully signed in',
45         accessToken,
46         refreshToken
47       });
48     } catch (error) {
49       console.error(error);
50       res.status(500).send('Internal server error');
51     }
52   });
});
```

**Εικόνα 7:** Διαδρομή για σύνδεση των χρηστών, όπου μετά την επαλήθευσή τους, δημιουργούνται ένα access και ένα refresh token και αποστέλλονται σε αυτούς (κώδικας από auth\_routes.js).

### 2.3.5.3 bcrypt

Το **bcrypt** είναι μια βιβλιοθήκη η οποία εξειδικεύεται στην κρυπτογράφηση κωδικών πρόσβασης και στην ασφαλή αποθήκευσή τους. Χρησιμοποιεί έναν αλγόριθμο κρυπτογράφησης που είναι ανθεκτικός σε επιθέσεις brute-force. Λειτουργεί δημιουργώντας ένα μοναδικό hash για κάθε κωδικό, ώστε σε περίπτωση διαρροής της βάσης δεδομένων, οι αρχικοί κωδικοί να μην είναι εύκολα ανακτήσιμοι. Το bcrypt συνδυάζει το hashing με μια τεχνική γνωστή ως salting [38], [39].

Με τον ίδιο τρόπο χρησιμοποιείται στην εφαρμογή μας για την κρυπτογράφηση κωδικών πρόσβασης κατά την εγγραφή νέων χρηστών, την αλλαγή κωδικού και για την επαλήθευση των κωδικών κατά τη σύνδεση. Έτσι, ο κωδικός που υποβάλλουν πρώτη φορά αποθηκεύεται διαφοροποιημένος στη βάση δεδομένων έτσι ώστε να μην υπάρχει επικινδυνότητα διαρροής του [38], [39]. Περιλαμβάνεται στα αρχεία διαδρομών που διαχειρίζονται την εγγραφή (user\_routes.js) και την αυθεντικοποίηση χρηστών (auth\_routes.js).

### 2.3.5.4 Helmet

Το **Helmet** είναι μια βιβλιοθήκη για το Express.js που προσθέτει επίπεδα ασφάλειας στις HTTP κεφαλίδες, προστατεύοντας τις εφαρμογές Node.js από κοινές επιθέσεις όπως Cross-Site Scripting (XSS), clickjacking και άλλες που εκμεταλλεύονται αδυναμίες του HTTP [40], [41].

Το **Helmet** χρησιμοποιείται μόνο στο αρχείο app.js για να βελτιώσει την ασφάλεια της εφαρμογής προσθέτοντας προστατευτικές HTTP κεφαλίδες στα αιτήματα από τη στιγμή που ξεκινά ο server. Αυτό μειώνει την πιθανότητα εκμετάλλευσης ευπαθειών και βελτιώνει τη συνολική ασφάλεια των χρηστών και των δεδομένων τους, λόγοι που οδήγησαν στην επιλογή του Helmet για την παροχή προστασίας στην εφαρμογή μας. Με την απλή προσθήκη της εντολής “`app.use(helmet());`” πραγματοποιούνται τα παραπάνω [40], [41].

### 2.3.5.5 Morgan

Το **Morgan** είναι ένα middleware logger για το Node.js (μέσω του Express.js), που χρησιμοποιείται για την καταγραφή των αιτημάτων HTTP που γίνονται στον server [22]. Χρησιμοποιείται κυρίως για debugging και παρακολούθηση της δραστηριότητας των αιτημάτων, καταγράφοντας πληροφορίες όπως τη μέθοδο, την κατάσταση και τη διαδρομή του αιτήματος [42], [43].

Στην εφαρμογή μας, το **Morgan** έχει ρυθμιστεί στο αρχείο app.js για την καταγραφή όλων των αιτημάτων που γίνονται προς τον server, βοηθώντας έτσι στην παρακολούθηση των ενεργειών των χρηστών και την ανίχνευση σφαλμάτων ή μη φυσιολογικής δραστηριότητας για debugging. Με την απλή προσθήκη της εντολής “`app.use(morgan('combined'));`” πραγματοποιούνται τα παραπάνω, χρησιμοποιώντας το “**combined**” format που προσφέρει εκτενέστερη καταγραφή, περιλαμβάνοντας πληροφορίες όπως τη διεύθυνση IP των χρηστών, την ημερομηνία και ώρα του αιτήματος, τη μέθοδο κ.α.[42], [43].

### 2.3.5.6 CORS

Το **CORS** (Cross-Origin Resource Sharing) είναι ένας μηχανισμός ασφαλείας που επιτρέπει ή περιορίζει αιτήματα HTTP μεταξύ διαφορετικών τομέων (domains) στο Express.js.

Χρησιμοποιείται για να επιτρέπει σε έναν server να δέχεται αιτήματα από έναν άλλο domain, κάτι που είναι απαραίτητο όταν το **Back-End** και το **Front-End** μιας εφαρμογής βρίσκονται σε διαφορετικά domains ή θύρες [44], [45].

Στην εφαρμογή μας, το **CORS** ρυθμίζεται μέσω της εντολής `app.use(cors())`; στο `app.js`. Αυτό επιτρέπει στο Back-End να εξυπηρετεί αιτήματα από το Front-End χωρίς προβλήματα ασφαλείας ή περιορισμούς προέλευσης. Η εντολή περιλαμβάνει ορισμένες παραμέτρους με κάθε μια να εξυπηρετεί συγκεκριμένους σκοπούς. Η παράμετρος **origin** επιτρέπει αιτήματα μόνο από το “`http://localhost:3001`” (Front-End), η **methods** καθορίζει τις επιτρεπόμενες μεθόδους HTTP, η **credentials** ενεργοποιεί την αποστολή cookies και πληροφοριών ταυτοποίησης και η **allowedHeaders** καθορίζει τα επιτρεπόμενα headers, δηλαδή τα “Content-Type” και “Authorization” [44], [45]. Η εντολή “`app.options('*', cors({...}))`” ρυθμίζει τα προαιρετικά αιτήματα για όλους τους προορισμούς (\*). Οι παράμετροι που περιλαμβάνει είναι οι αντίστοιχες της παραπάνω εντολής. Όσα αναλύθηκαν παρατηρούνται στην εικόνα οκτώ.

```
29 // Enable CORS
30 app.use(cors({
31   origin: 'http://localhost:3001',
32   methods: ['GET', 'POST', 'PUT', 'DELETE'],
33   credentials: true,
34   allowedHeaders: ['Content-Type', 'Authorization'],
35 }));
36
37 app.options('*', cors({
38   origin: 'http://localhost:3001',
39   methods: ['GET', 'POST', 'PUT', 'DELETE'],
40   allowedHeaders: ['Content-Type', 'Authorization'],
41   credentials: true
42 }));
```

*Εικόνα 8: Κύριος κώδικας χρήσης CORS στο αρχείο `app.js` του Back-End της εφαρμογής.*

### 2.3.5.7 Router

Το **Router** στο `Express.js` είναι ένα υποσύστημα που επιτρέπει τη δημιουργία και τη διαχείριση των διαδρομών (routes) της εφαρμογής. Είναι ένα εργαλείο που επιτρέπει τη δρομολόγηση αιτημάτων HTTP σε συγκεκριμένα τμήματα κώδικα ανάλογα με την καθορισμένη διαδρομή και μέθοδο (GET, POST, PUT, DELETE, κ.λπ.) [22], [32]. Η χρήση του διευκολύνει τη διαχείριση της λογικής δρομολόγησης της εφαρμογής, επιτρέποντας την

οργάνωση των routes σε διαφορετικά αρχεία και την ανάθεσή τους σε ξεχωριστά κομμάτια λειτουργικότητας [46], [47].

Στην εφαρμογή, το **Router** χρησιμοποιείται για να διαχωρίσει τη λογική των διαδρομών (routes) ανάλογα με τη λειτουργικότητα, δηλαδή για τη διαχείριση ρούχων, συνόλων, περιστάσεων, ταξιδιωτικών λιστών, και χρηστών. Κάθε λειτουργία που εκτελείται για αυτά τα στοιχεία έχει τις αντίστοιχες διαδρομές που είναι ορισμένες με το Router [46], [47]. Χρησιμοποιείται στα αρχεία διαδρομών όπως αυτά για τη διαχείριση των ρούχων, συνόλων, περιστάσεων, ταξιδιωτικών λιστών και χρηστών. Επίσης, χρησιμοποιείται στο αρχείο “auth\_routes.js” για τη διαχείριση της αυθεντικοποίησης χρηστών. Σε κάθε αρχείο ο ρόλος του είναι να δρομολογεί αιτήματα και να αντιστοιχεί συγκεκριμένες λειτουργίες, εξασφαλίζοντας ότι οι κατάλληλες λειτουργίες εκτελούνται όταν λαμβάνονται αιτήματα από το Front-End.

```
131 // Delete the authenticated user's account
132 router.delete('/me', authenticateToken, async (req, res) => {
133   try {
134     const user = await User.findByIdAndDelete(req.user.id);
135     if (!user) {
136       return res.status(404).json({ message: 'User not found' });
137     }
138     res.json({ message: 'User account deleted successfully' });
139   } catch (err) {
140     res.status(500).json({ message: 'Internal server error' });
141   }
142 });


```

**Εικόνα 9:** Τμήμα κώδικα που επιτρέπει στους αυθεντικοποιημένους χρήστες να διαγράφουν τον λογαριασμό τους από τη βάση δεδομένων με τη μέθοδο DELETE (από το αρχείο user\_routes.js).

### 2.3.5.8 dotenv

Το **dotenv** είναι μια βιβλιοθήκη για το Node.js που επιτρέπει τη διαχείριση των μεταβλητών περιβάλλοντος σε αρχεία **.env**. Αυτά τα αρχεία περιέχουν κλειδιά και τιμές που χρησιμοποιούνται στην εφαρμογή, όπως διαπιστευτήρια σύνδεσης με τη βάση δεδομένων ή μυστικά κλειδιά (secrets). Η χρήση του dotenv επιτρέπει τη διαχείριση και ασφάλεια των ευαίσθητων πληροφοριών, διαχωρίζοντάς τα από τον κώδικα της εφαρμογής [48], [49].

Στην εφαρμογή μας, το **dotenv** χρησιμοποιείται στο αρχείο **app.js** για να φορτώνει τις μεταβλητές που περιέχουν ευαίσθητες πληροφορίες, όπως το URI της MongoDB και τα JWT

Secret Keys [28], [30]. Αυτά αποθηκεύονται στο αρχείο .env, το οποίο δεν περιλαμβάνεται στον κώδικα για λόγους ασφάλειας, αλλά παραμένει προσβάσιμο στο μέσω του “process.env”. Με τη συμπερίληψη της εντολής “`require('dotenv').config();`” στο app.js, φορτώνονται οι μεταβλητές περιβάλλοντος που βρίσκονται στο αρχείο .env.

### 2.3.5.9 body-parser

Το **body-parser** είναι ένα middleware για το Express.js που επιτρέπει την ανάλυση (parsing) των δεδομένων που περιέχονται στο σώμα ενός HTTP αιτήματος [22]. Συγκεκριμένα, μετατρέπει τα δεδομένα σε αντικείμενα JavaScript για εύκολη πρόσβαση και επεξεργασία μέσα στον κώδικα. Μπορεί να αναλύει δεδομένα σε μορφή JSON, χρησιμοποιώντας τη μέθοδο bodyParser.json(), ενώ επίσης μπορεί να αναλύει δεδομένα που αποστέλλονται μέσω HTML forms, χρησιμοποιώντας τη μέθοδο bodyParser.urlencoded() [50], [51].

Στην εφαρμογή, το **body-parser** χρησιμοποιείται στο app.js για την ομαλή λήψη και επεξεργασία των δεδομένων που αποστέλλονται από αιτήματα του Front-End στις διαδρομές του Back-End, διευκολύνοντας την αλληλεπίδραση μεταξύ τους. Με τη χρήση των εντολών “`app.use(bodyParser.json());`” και “`app.use(bodyParser.urlencoded({ extended: true }));`” επιτρέπεται στην εφαρμογή να λαμβάνει δεδομένα σε μορφή JSON ή URL-encoded και να τα μετατρέπει σε αντικείμενα JavaScript, τα οποία μπορούν να χρησιμοποιηθούν απευθείας στον κώδικα.

### 2.3.5.10 express-validator

Το **express-validator** είναι μια δημοφιλής βιβλιοθήκη επαλήθευσης δεδομένων για το Express.js. Παρέχει μια σειρά από middleware που βοηθούν στην επικύρωση και τον καθαρισμό των δεδομένων εισόδου, από ανεπιθύμητους χαρακτήρες, από τους χρήστες. Χρησιμοποιεί κανόνες επαλήθευσης που εφαρμόζονται στα αιτήματα HTTP και προσφέρει μεθόδους για την αντιμετώπιση και αναφορά σφαλμάτων [52], [53].

Στην εφαρμογή, το **express-validator** επαληθεύει τα δεδομένα που αποστέλλονται από τους χρήστες για ενέργειες όπως η εγγραφή και η σύνδεση. Συγκεκριμένα, ελέγχει τα δεδομένα, διασφαλίζοντας ότι οι τιμές που εισάγονται οι χρήστες είναι έγκυρες, όπως email, κωδικός πρόσβασης με συγκεκριμένα χαρακτηριστικά, κ.λπ. Επιπλέον, καθαρίζει τα δεδομένα, αφαιρώντας ανεπιθύμητους χαρακτήρες ή κενά που μπορεί να έχουν εισαχθεί κατά λάθος από

τους χρήστες. Χρησιμοποιείται σε όλα τα αρχεία διαδρομών (routes), με την εντολή “**const { body, validationResult } = require('express-validator');**”, προσφέροντας μια δομή για την υλοποίηση όλων των βασικών λειτουργιών της εφαρμογής.

### 2.3.5.11 Path

Το **path** είναι μια ενσωματωμένη βιβλιοθήκη του Node.js που παρέχει εργαλεία για τη διαχείριση και την επίλυση μονοπατιών αρχείων και φακέλων στο σύστημα αρχείων. Παρέχει διάφορες μεθόδους για τη δημιουργία, κανονικοποίηση, σύνδεση και ανάλυση διαδρομών αρχείων και φακέλων, προσφέροντας ευελιξία και ασφάλεια κατά την εργασία με τοπικά συστήματα αρχείων σε εφαρμογές Node.js [54].

Στην εφαρμογή, ο ρόλος του **path** είναι να χειρίζεται τις διαδρομές αρχείων όταν γίνεται ανέβασμα αυτών ή δημιουργία και αναφορά σε καταλόγους. Συγκεκριμένα, εξασφαλίζει ότι οι εικόνες αποθηκεύονται στις σωστές τοποθεσίες στο server, μέσα στους καταλόγους uploads/pfp και uploads/clothes. Περιλαμβάνεται στα αρχεία διαδρομών όπου γίνεται διαχείριση των αποστολών αρχείων. Εισάγεται στα αρχεία app.js και auth\_routes.js με την εντολή “**const path = require('path');**”.

```
51
52 // Serve static files from the uploads directory
53 app.use('/uploads/pfp', express.static(path.join(__dirname, 'uploads/pfp')));
54 app.use('/uploads/clothes', express.static(path.join(__dirname, 'uploads/clothes')));
55
```

**Εικόνα 10:** Τμήμα κώδικα του app.js όπου διακρίνεται η χρήση του Path για την εξυπηρέτηση στατικών αρχείων από τα uploads/pfp και uploads/clothes.

## 2.4 Υλοποίηση και Κώδικας

Σε αυτή την ενότητα, θα περιγραφεί όλη η υλοποίηση του Back-End της εφαρμογής "Digital Wardrobe". Συγκεκριμένα, θα εστιάσουμε στην ακολουθία των βημάτων δημιουργίας, τα κυριότερα εργαλεία που χρησιμοποιήθηκαν, τις μεθόδους που εφαρμόστηκαν για την ανάπτυξη και δοκιμή της εφαρμογής και τα σημαντικότερα τμήματα κώδικα που εξυπηρετούν τα κύρια ζητούμενα της διαδικτυακής εφαρμογής.

## 2.4.1 Προετοιμασία Περιβάλλοντος Ανάπτυξης

Έπειτα από εκτενή αναζήτηση για τις κατάλληλες βάσεις δεδομένων οι οποίες είναι ευέλικτες για διαδικτυακές εφαρμογές καταλήξαμε στη MongoDB την οποία εγκαταστήσαμε από την επίσημη ιστοσελίδα της. Έγινε αρκετή εξάσκηση για εξοικείωση με το εργαλείο μέσω των MongoDB Compass χειροκίνητα και Mongo Shell (μέσω CMD) με τη δοκιμή πολλών διαφορετικών εντολών για τη δημιουργία συλλογών και εγγράφων, εύρεση ή διαγραφή τους κ.α. [26]. Οι εντολές βρέθηκαν έπειτα από αναζήτηση στο διαδίκτυο και η εκμάθηση του εργαλείου μέσω εκπαιδευτικών βίντεο στο YouTube.

Το επόμενο βήμα ήταν η εγκατάσταση της Node.js από την επίσημη ιστοσελίδα του και η δημιουργία ενός αρχικού φακέλου έργου με την ονομασία “Digital Wardrobe”. Έπειτα εγκαταστάθηκε το Express.js στον φάκελο με την εντολή “**npm install express**” στη γραμμή εντολών (CMD) και δημιουργήθηκε ο server [25]. Στο αρχείο app.js δημιουργήθηκε η σύνδεση με τη βάση δεδομένων η οποία τρέχει στο port 3000.

## 2.4.2 Ανάπτυξη και Δημιουργία Μοντέλων

Με την εγκατάσταση της βιβλιοθήκης Mongoose (“**npm install mongoose**”), αναπτύχθηκαν τα μοντέλα για τα βασικά αντικείμενα της εφαρμογής, όπως τα ρούχα, σύνολα, περιστάσεις, λίστες ταξιδιών και χρήστες. Καθένα από αυτά δημιουργήθηκε εντός του φακέλου “models” και ορίστηκε με βάση τα σχήματα (schemas) που περιέχουν τα κατάλληλα πεδία, τους τύπους δεδομένων και τις επικυρώσεις που απαιτούνται για τη σωστή διαχείριση δεδομένων. Επιλέχθηκαν πολύ βασικά πεδία για κάθε μοντέλο, με κάποια από αυτά να είναι υποχρεωτικά στη συμπλήρωση καθώς έχουν κεντρικό ρόλο στη λειτουργία της εφαρμογής [21]. Παρακάτω (εικόνα 11) φαίνεται το μοντέλο των χρηστών.

```
1 const mongoose = require('mongoose');
2
3 const userSchema = new mongoose.Schema({
4   email: { type: String, required: [true, 'E-mail is required'], unique: true },
5   password: { type: String, required: [true, 'Password is required'] },
6   name: { type: String, required: [true, 'Please type your name'] },
7   surname: { type: String, required: [true, 'Please type your surname'] },
8   age: {type: Number, default: null },
9   favorite_colors: { type: [String], default: [] },
10  profile_picture_url: { type: String, default: null }
11 }, { collection: "Users" });
12
13 const User = mongoose.model('User', userSchema);
14 module.exports = User;
```

**Εικόνα 11:** Το μοντέλο των χρηστών στο Back-End με όλα τα απαραίτητα πεδία και τα προαιρετικά (αρχείο “users.js” στον φάκελο models).

### 2.4.3 Διαχείριση Εικόνων και Ανάπτυξη Διαδρομών

Μετά την ολοκλήρωση των μοντέλων, δημιουργήσαμε έναν φάκελο εντός του “Digital Wardrobe” με την ονομασία “uploads”, όπου αποθηκεύονται οι εικόνες προφίλ των χρηστών και των ρούχων. Καθορίσαμε διαφορετικούς υποφακέλους για τις εικόνες προφίλ (uploads/pfp) και για τις εικόνες των ρούχων (uploads/clothes) ώστε να είναι οργανωμένες ξεχωριστά.

Στη συνέχεια, αναπτύξαμε μια-μια τις διαδρομές (routes) για κάθε βασική λειτουργία της εφαρμογής. Σε κάθε διαδρομή συμπεριλάβαμε τις απαραίτητες βιβλιοθήκες που εξυπηρετούν τις ανάγκες της εφαρμογής και την ίδια λογική πίσω από κάθε τμήμα κώδικα, όπως αναλύθηκε στην παραπάνω ενότητα. Κάθε διαδρομή ορίστηκε με το Express.js για να διαχειρίζεται τα HTTP αιτήματα (GET, POST, PUT, DELETE) που αποστέλλονται από το Front-End όταν οι χρήστες εκτελούν κάποια σχετική ενέργεια (δημιουργία, επεξεργασία, ανάκτηση και διαγραφή) [22]. Οι διαδρομές των ρούχων και των χρηστών συνδέονται με τους υποφακέλους του “uploads” ώστε να αποθηκεύονται οι εικόνες σε αυτούς. Στην παρακάτω εικόνα διακρίνεται η δημιουργία ενός νέου χρήστη (POST) με όλα τα απαραίτητα πεδία, και οι βιβλιοθήκες, όπως path, bcrypt, που ολοκληρώνουν την ενέργεια.

```

22 // POST route to register a new user with optional profile picture
23 router.post('/register', upload.single('profile_picture'), [
24   body('email').isEmail().withMessage('Invalid email address'),
25   body('password').isLength({ min: 6 }).withMessage('Password must be at least 6 characters long'),
26   body('name').not().isEmpty().withMessage('Name is required'),
27   body('surname').not().isEmpty().withMessage('Surname is required')
28 ], async (req, res) => {
29   const errors = validationResult(req);
30   if (!errors.isEmpty()) {
31     return res.status(400).json({ errors: errors.array() });
32   }
33
34   const hashedPassword = await bcrypt.hash(req.body.password, 10);
35   const user = new User({
36     email: req.body.email,
37     password: hashedPassword,
38     name: req.body.name,
39     surname: req.body.surname,
40     age: req.body.age || null,
41     favorite_colors: req.body.favorite_colors || [],
42     profile_picture_url: req.file ? req.file.path : null
43   });
44
45   try {
46     const newUser = await user.save();
47     res.status(201).json(newUser);
48   } catch (err) {
49     res.status(400).json({ message: err.message });
50   }
51 });

```

**Εικόνα 12:** Μέθοδος *POST* για τη διαδρομή των χρηστών με όλα τα πεδία συμπλήρωσης και σημαντικές βιβλιοθήκες (από το αρχείο *user\_routes.js*).

#### 2.4.4 Δημιουργία και Διαχείριση Αυθεντικοποίησης

Το τελευταίο βήμα ολοκλήρωσης ανάπτυξης του Back-End έγινε με τη δημιουργία του συστήματος αυθεντικοποίησης, το οποίο σχεδιάστηκε για να προσφέρει ένα υψηλό επίπεδο ασφάλειας και προστασίας στα δεδομένα των χρηστών. Ο κεντρικός σκοπός του συστήματος είναι να διασφαλίσει ότι μόνο οι αυθεντικοποιημένοι χρήστες μπορούν να έχουν πρόσβαση σε όλες τις λειτουργίες της εφαρμογής. Η υλοποίηση εστιάστηκε στη χρήση Access και Refresh Tokens, μέσω της βιβλιοθήκης jsonwebtoken (JWT) για τη δημιουργία και επαλήθευση αυτών [30], [37]. Η χρήση της βιβλιοθήκης αναλύθηκε περαιτέρω στην προηγούμενη ενότητα (2.3.5.2).

Δημιουργήθηκαν τρία αρχεία που αφορούν την αυθεντικοποίηση, με πρώτο το “**token.js**” κατά το οποίο χρησιμοποιούνται συναρτήσεις όπως το “**generateAccessToken()**” και το “**generateRefreshToken()**”, οι οποίες δημιουργούν τα tokens με καθορισμένη διάρκεια ζωής. Το Access Token έχει διάρκεια ζωής μιας ώρας, ενώ το Refresh Token ισχύει για επτά ημέρες (Εικόνα 13). Κατά τη σύνδεση στο προφίλ το σύστημα ελέγχει τα διαπιστευτήριά των

χρηστών και, αν είναι έγκυρα, δημιουργούνται τα tokens και επιστρέφονται ως απάντηση από τον διακομιστή.

```
6  const generateAccessToken = (user) => {
7    |   return jwt.sign({ id: user._id, type: 'access' }, accessTokenSecret, { expiresIn: '1h' });
8  };
9
10 const generateRefreshToken = (user) => {
11   |   return jwt.sign({ id: user._id, type: 'refresh' }, refreshTokenSecret, { expiresIn: '7d' });
12 };
```

**Εικόνα 13:** Συναρτήσεις δημιουργίας των δυο tokens στο αρχείο token.js.

Ταυτόχρονα, σε ξεχωριστό αρχείο με όνομα “**authenticateToken.js**”, μέσω της συνάρτησης “authenticateToken” πραγματοποιείται η επαλήθευση των tokens που συνοδεύουν κάθε αίτημα των χρηστών. Εφόσον το Access Token είναι έγκυρο, αποθηκεύεται το αντικείμενο των χρηστών στο αίτημα (req.user), επιτρέποντας την πρόσβαση σε όλες τις λειτουργίες.

Σε ένα τρίτο αρχείο με όνομα “**auth\_routes.js**”, υλοποιούνται οι διαδρομές (routes) που αφορούν τη διαδικασία αυθεντικοποίησης και εγγραφής των χρηστών της εφαρμογής Digital Wardrobe. Συγκεκριμένα, περιλαμβάνει διαδρομή **/signin**, όπου οι χρήστες μπορούν να συνδεθούν παρέχοντας email και κωδικό πρόσβασης. Με την επιτυχή επαλήθευσή τους, δημιουργούνται τα Access και Refresh Token, τα οποία επιστρέφονται για την πιστοποίησή τους στις βασικές λειτουργίες της εφαρμογής. Επίσης, περιλαμβάνει διαδρομή **/signup** για την εγγραφή νέων χρηστών, όπου τα στοιχεία τους αποθηκεύονται με ασφάλεια στη βάση δεδομένων, κρυπτογραφώντας τους κωδικούς πρόσβασης με την bcrypt [38], [39]. Τέλος, περιέχει διαδρομές για την ανανέωση των tokens και τη διαχείριση της αποσύνδεσης, εξασφαλίζοντας την ασφαλή διαχείριση της πρόσβασης των χρηστών στις υπηρεσίες της εφαρμογής.

```
30   router.post('/signin', async (req, res) => {
31     try {
32       const { email, password } = req.body;
33       const user = await User.findOne({ email });
34       if (!user) {
35         return res.status(401).send('Invalid email or password');
36       }
37       const isMatch = await bcrypt.compare(password, user.password);
38       if (!isMatch) {
39         return res.status(401).send('Invalid email or password');
40       }
41       const accessToken = generateAccessToken(user);
42       const refreshToken = generateRefreshToken(user);
43       res.json({
44         message: 'Successfully signed in',
45         accessToken,
46         refreshToken
47       });
48     } catch (error) {
49       console.error(error);
50       res.status(500).send('Internal server error');
51     }
52   });
});
```

**Εικόνα 14:** Διαδρομή σύνδεσης στην εφαρμογή που πραγματοποιείται στο αρχείο διαδρομής αυθεντικοποίησης και εγγραφής των χρηστών (*auth\_routes.js*).

## 2.4.5 Έλεγχος, δοκιμή και προσαρμογές για το Front-End

Κατά τη διάρκεια της ανάπτυξης αλλά και προς την ολοκλήρωσή του Back-End, θέταμε σε λειτουργία τον server και τη σύνδεση με τη βάση δεδομένων με την εντολή “**node app.js**” στη γραμμή εντολών (cmd) [24]. Παράλληλα, δημιουργήθηκε ένας ξεχωριστός φάκελος εκτός του Back-End, που αφορά το Front-End, μέσα στον οποίο γίνεται χρήση της React.js [16]. Αν και θα αναλυθεί η διαδικασία δημιουργίας του Front-End στο επόμενο κεφάλαιο, αξίζει να σημειωθεί ότι στο πρώιμο στάδιό του, θέταμε σε λειτουργία την εφαρμογή με την εντολή “**npm start**”, κατά την οποία έτρεχε και άνοιγε μια απλή και αδόμητη σχεδιαστικά σελίδα στο περιηγητή ιστού. Εκεί πραγματοποιούνταν ο έλεγχος ομαλής λειτουργίας και επικοινωνίας του Back-End με το Front-End, αλλά και δοκιμές στις βασικές λειτουργίες της εφαρμογής.

Οι πίνακες “Console” και “Network” του περιηγητή ιστού, προσβάσιμοι μέσω του “Inspect Element” (έλεγχος) συνέβαλαν στον εντοπισμό σφαλμάτων, ελλείψεων και προβλημάτων προς διόρθωση. Επομένως, η ολοκλήρωση του Back-End πραγματοποιήθηκε με τις κρίσιμες προσαρμογές βάση των αναγκών του Front-End, κάνοντας τις απαραίτητες αλλαγές για να διασφαλιστεί η σωστή αλληλεπίδραση μεταξύ των δύο βασικών τμημάτων της εφαρμογής. Οι

προσαρμογές που έγιναν αλλά και η ακολουθία βημάτων ανάπτυξης του Front-End θα αναλυθούν στο επόμενο κεφάλαιο

## 2.4.6 Εργαλεία Υποστήριξης

Για την ανάπτυξη του Back-End της εφαρμογής, χρησιμοποιήθηκαν πολλά σημαντικά εργαλεία που υποστήριξαν την ανάπτυξη του κώδικα αλλά και ενέπνευσαν την ιδεολογία και μεθοδολογία που ακολουθήθηκε. Αρχικά, η εκμάθηση των γλωσσών προγραμματισμού αλλά και η εισαγωγή στα εργαλεία που χρησιμοποιήθηκαν πραγματοποιήθηκε έπειτα από αναζήτηση στο διαδίκτυο αλλά και στην τεχνητή νοημοσύνη του ChatGPT. Συγκεκριμένα, εκπαιδευτικά βίντεο στο **YouTube**, αλλά και ιστοσελίδες όπως **Stack Overflow** και **W3Schools**, συνέβαλαν στη δομή των γνώσεων γύρω από την ανάπτυξη του Back-End έπειτα από αναλυτική έρευνα πάνω στις τεχνολογίες που επιλέχθηκαν. Το **ChatGPT** συνέβαλε επίσης σημαντικά στην επεξήγηση δυσνόητων, αρχικά, τμημάτων κώδικα και λειτουργιών, στην πρόταση βελτιώσεων αλλά και στην αποσφαλμάτωση για όσα δεν βρισκόταν κάποια αποτελεσματική λύση.

Για την εξάσκηση με τη βάση δεδομένων χρησιμοποιήθηκε τόσο το περιβάλλον **MongoDB Compass** όσο και το **Mongo Shell** στη γραμμή εντολών, στα οποία δοκιμάστηκαν πολλές απλές και σύνθετες εντολές προκειμένου να επιτευχθεί η εξοικείωση με τη βάση δεδομένων [14]. Όλοι οι κώδικες που δημιουργήθηκαν για την πλήρη στοίχιση του Back-End όπως τα μοντέλα, οι διαδρομές και το κεντρικό αρχείο app.js, αναπτύχθηκαν στο περιβάλλον ανάπτυξης κώδικα **Visual Studio Code** και η εξέταση λειτουργίας αυτών αλλά και της σύνδεσης με τη βάση δεδομένων έγινε με τη βοήθεια της **γραμμής εντολών (CMD)**.

## 2.5 Προκλήσεις και Εμπόδια

Κατά την πορεία ανάπτυξης του Back-End και της βάσης δεδομένων εμφανίστηκαν και αντιμετωπίστηκαν διάφορες προκλήσεις και εμπόδια, κυρίως τεχνικά. Αυτά αποτέλεσαν η ασυγχρονία και διαχείριση πολλαπλών αιτημάτων, η διαχείριση επικύρωσης δεδομένων και σφαλμάτων, η πολυπλοκότητα στη δημιουργία των μοντέλων, πρόβλημα σύνδεσης με τη βάση δεδομένων, η αναβάθμιση βιβλιοθηκών και η διαχείριση αποθηκευτικών μέσων. Θα αναλυθούν παρακάτω, ενώ επίσης θα παρουσιαστούν λύσεις/βελτιώσεις που λήφθηκαν για την αντιμετώπισή τους.

## 2.5.1 Προκλήσεις και Λύσεις

Καθώς το Back-End της εφαρμογής μας καλείται να χειρίζεται πολλά αιτήματα ταυτόχρονα από διάφορους χρήστες, η ανάγκη για ασύγχρονες λειτουργίες με το Node.js μπορεί να δημιουργήσει δυσκολίες στη διαχείριση της αλληλουχίας και των απαντήσεων. Αυτό θα μπορούσε να έχει ως αποτέλεσμα την ασυνέπεια στην ενημέρωση των δεδομένων ή καθυστερήσεις στις αποκρίσεις. Για να αποφευχθεί αυτό, χρησιμοποιήθηκαν ασύγχρονες συναρτήσεις **Async & Await**, λέξεις-κλειδιά που αντικαθιστούν τις παραδοσιακές προσεγγίσεις callbacks και promises, σε όλες τις βασικές λειτουργίες, ώστε να εξασφαλιστεί η σωστή εκτέλεση των αιτημάτων και η μείωση των σφαλμάτων που προκύπτουν από την ασυγχρονία.

Η επικύρωση των δεδομένων και η διαχείριση των σφαλμάτων αποτέλεσε κρίσιμο σημείο κατά την ανάπτυξη. Έπρεπε να διασφαλιστεί ότι οι χρήστες εισάγουν μόνο έγκυρα δεδομένα, ειδικά κατά την προσθήκη και επεξεργασία ρούχων ή προφίλ. Για αυτό τον λόγο χρησιμοποιήθηκε το **Express Validator**, κατάλληλο για την επικύρωση δεδομένων, ενώ παράλληλα γράφτηκαν κατάλληλοι χειρισμοί σφαλμάτων για την ανατροφοδότηση στους χρήστες ώστε να προσέξουν τα λάθη τους και να επαναλάβουν την κίνησή τους.

Κατά τη δημιουργία των μοντέλων ρούχων, συνόλων, περιστάσεων, ταξιδιωτικών λιστών και χρηστών με το Mongoose, προέκυψαν ζητήματα σχετικά με την επιλογή και την οργάνωση των πεδίων, ειδικά όταν κάποια από αυτά απαιτούν την αποθήκευση πολλαπλών τιμών. Έτσι έγινε μια αναδιοργάνωση στα μοντέλα, βελτιώνοντας τη σύνδεση μεταξύ τους με τη **χρήση αναγνωριστικών (ids)** χρηστών, ρούχων και συνόλων για να εξασφαλιστεί η σωστή αναφορά μεταξύ των διαφορετικών συλλογών.

## 2.5.2 Τεχνικά Εμπόδια και Λύσεις

Κατά τη σύνδεση της εφαρμογής με τη MongoDB και τη χρήση της βιβλιοθήκης Mongoose, υπήρχαν προκλήσεις με το URI σύνδεσης και την ασφάλεια. Οι διακοπές στη σύνδεση ή η λανθασμένη διαμόρφωση του περιβάλλοντος οδήγησαν σε σφάλματα που χρειάστηκε να εντοπιστούν και να διορθωθούν. Για αυτό το λόγο προστέθηκαν συγκεκριμένες παράμετροι στο URI, όπως “**useNewUrlParser: true**” και “**useUnifiedTopology: true**”, για την αποφυγή ασυμβατότητας και για τη διασφάλιση σταθερής σύνδεσης. Εφαρμόστηκαν μηχανισμοί κρυπτογράφησης κωδικών και βελτιώθηκε η διαχείριση credentials μέσω περιβαλλοντικών μεταβλητών χρησιμοποιώντας τη βιβλιοθήκη **dotenv**.

Κατά την ανάπτυξη, η παράλειψη ενημέρωσης των βιβλιοθηκών, όπως το Express.js και το Mongoose, οδήγησαν σε μερικές ασυμβατότητες και προβλήματα στους ανεπτυγμένους κώδικες. Αυτές οι ασυμβατότητες απαιτούσαν την εγκατάσταση των πιο πρόσφατων ενημερώσεων καθώς μόνο με αυτό τον τρόπο θα αναγνωρίζονταν όλες οι ανεπτυγμένες λειτουργίες. Οι ενημερώσεις πραγματοποιήθηκαν με την εντολή “**npm install ..**” στη γραμμή εντολών, συνοδευόμενη από την ονομασία της βιβλιοθήκης.

Η αποθήκευση των εικόνων των ρούχων και των προφίλ σε φακέλους (/uploads) εντός της εφαρμογής αποτέλεσε πρόκληση. Έπρεπε να εξασφαλιστεί η σωστή αποθήκευση και πρόσβαση στις εικόνες και να αντιμετωπιστούν ζητήματα ασφάλειας, όπως η μη εξουσιοδοτημένη πρόσβαση σε ευαίσθητα δεδομένα. Για τη σωστή αποθήκευση χρησιμοποιήθηκε η βιβλιοθήκη **Multer** που αναλαμβάνει τη μεταφόρτωση αρχείων σε συγκεκριμένους προορισμούς, ενώ για τον έλεγχο πρόσβασης εφαρμόστηκαν πολιτικές στις διαδρομές που διασφαλίζουν ότι μόνο εξουσιοδοτημένοι χρήστες μπορούν να προβάλλουν ή να τροποποιήσουν τα αρχεία.

## 2.6 Συμπεράσματα

Η ανάπτυξη του Back-End του Digital Wardrobe είχε ως κύριους στόχους την αποδοτικότητα, την επεκτασιμότητα και την ασφάλεια των δεδομένων των χρηστών. Οι τεχνολογίες και τα εργαλεία που επιλέχθηκαν, όπως το Node.js, το Express.js και το MongoDB, αποδείχθηκαν κατάλληλα για την ανάπτυξη ενός δυναμικού και ευέλικτου συστήματος που καλύπτει όλους τους στόχους που είχαν τεθεί για τη διαδικτυακή εφαρμογή.

Κατά τη δόμησή του, αντιμετωπίστηκαν πολλά τεχνολογικά εμπόδια και προκλήσεις. Ωστόσο, η χρήση απαραίτητων βιβλιοθηκών, η τμηματική σχεδίαση και η αναγνώριση των τεχνικών απαιτήσεων συνέβαλαν στην επιτυχή αντιμετώπισή τους και εύρεση λύσεων. Εργαλεία/βιβλιοθήκες όπως το Mongoose, το bcrypt και το Multer, μαζί με τα πλήρως δομημένα αρχεία διαδρομών και των μοντέλων όσον αφορά την ορθότητα διαχείρισης, προσέφεραν ένα ισχυρό και λειτουργικό αποτέλεσμα.

Το Back-End ολοκληρώθηκε και προσαρμόστηκε με βάση τις ανάγκες του Front-End, διασφαλίζοντας την ομαλή επικοινωνία και την αλληλουχία λειτουργιών μεταξύ τους. Η σύνδεση της βάσης δεδομένων, η διαχείριση των αιτημάτων και η εξουσιοδοτημένη δημιουργία και επεξεργασία των δεδομένων κατέστησαν δυνατή τη συνεργασία μεταξύ των δύο σημαντικών τμημάτων της εφαρμογής. Έτσι το Back-End ήταν πλέον έτοιμο να υποστηρίξει την ανάπτυξη του Front-End, που θα αναλυθεί στο επόμενο κεφάλαιο.

Βιβλιοθήκες που επιλέχθηκαν, όπως το Helmet για την ασφάλεια των κεφαλίδων και το jsonwebtoken για τη διαχείριση των tokens, βοήθησαν στην ενίσχυση της ασφάλειας των δεδομένων των χρηστών και στη δημιουργία ενός ασφαλούς περιβάλλοντος.

## 3 FRONT-END

Στο κεφάλαιο αυτό, θα αναλύσουμε τη δομή και τον ρόλο της λειτουργίας του Front-End της διαδικτυακής εφαρμογής “Digital Wardrobe”. Συγκεκριμένα, θα παρουσιαστούν τα κύρια εργαλεία, οι τεχνολογίες και οι βιβλιοθήκες που χρησιμοποιήθηκαν, τους λόγους επιλογής αυτών και πως κάθε μια από αυτές συμβάλλουν στην ανάπτυξη μιας φιλικής, απλής και διαδραστικής διεπαφής για τους χρήστες της εφαρμογής [56]. Επιπλέον, θα αναλυθούν όλα τα βήματα υλοποίησης με τη σειρά που ακολουθήθηκαν με τη συνοδεία στιγμιοτύπων σημαντικών τμημάτων κώδικα αλλά και της μορφής της διαδικτυακής εφαρμογής. Θα εξεταστεί κάθε περιεχόμενο της, με κύρια εστίαση να δίνεται στις βασικές λειτουργίες. Τέλος, θα γίνει αναφορά στις δοκιμές και τους ελέγχους που πραγματοποιήθηκαν καθ' όλη την πορεία υλοποίησης, καθώς και στις προκλήσεις και τα τεχνικά εμπόδια που αντιμετωπίσαμε κατά την ανάπτυξη του Front-End.

### 3.1 Εισαγωγή στο Front-End

Το Front-End της διαδικτυακής εφαρμογής είναι υπεύθυνο για την παρουσίαση και την αλληλεπίδραση των χρηστών με κάθε παρεχόμενη λειτουργία. Ο κύριος σκοπός του είναι να προσφέρει μια φιλική και εύχρηστη διεπαφή προς όλους τους χρήστες, επιτρέποντάς τους να δημιουργούν, να επεξεργάζονται και να διαγράφουν ρούχα, σύνολα, περιστάσεις, ταξιδιωτικές λίστες και προφίλ [56]. Επιπλέον, επικοινωνεί συνεχώς με το Back-End και τη βάση δεδομένων, στέλνοντας αιτήματα και λαμβάνοντας ως απάντηση δεδομένα με άμεσο και ομαλό τρόπο [28]. Επομένως, μέσω του Front-End, οι χρήστες μπορούν να έχουν πλήρη πρόσβαση σε όλες τις λειτουργίες της εφαρμογής, ενώ διασφαλίζεται η εμπειρία τους να είναι απλή, ευχάριστη και αποτελεσματική, δίνοντας ένα θετικό ξεκίνημα στην ημέρα τους.

#### 3.1.1 Ρόλος στην εφαρμογή

Ο κεντρικός ρόλος του Front-End της εφαρμογής “Digital Wardrobe” είναι να παρέχει στους χρήστες μια εύχρηστη και διαδραστική διεπαφή, μέσα από την οποία μπορούν να αλληλεπιδρούν με όλες τις λειτουργίες της εφαρμογής. Αυτές οι λειτουργίες περιλαμβάνουν για παράδειγμα τη δημιουργία και επεξεργασία προφίλ χρήστη, την προσθήκη και επεξεργασία ρούχων και τη διαγραφή συνόλων, περιστάσεων και ταξιδιωτικών λιστών. Το Front-End επικοινωνεί με το Back-End και τη MongoDB, στέλνοντας αιτήματα για την αποθήκευση, την επεξεργασία, την ανάκτηση ή τη διαγραφή δεδομένων και λαμβάνοντας τις

ανάλογες απαντήσεις με τις αλλαγές που πραγματοποιούνται [28]. Χρησιμοποιώντας το React.js, ιδανικό για τη δημιουργία ισχυρών και δυναμικών διεπαφών χρήστη [16], [57], το Front-End διασφαλίζει την άμεση και ομαλή ανανέωση των δεδομένων στις οθόνες των χρηστών, παρέχοντας μια απλή και απρόσκοπτη εμπειρία χρήστης.

### 3.1.2 Επικοινωνία με το Back-End και τη MongoDB

Στη διαδικτυακή μας εφαρμογή, το Front-End επικοινωνεί με το Back-End και, κατ' επέκταση, με τη βάση δεδομένων MongoDB μέσω HTTP αιτημάτων [22], χρησιμοποιώντας το Axios [58], του οποίου η λειτουργία θα αναλυθεί σε επόμενες ενότητες. Το Front-End στέλνει αιτήματα (GET, POST, PUT, DELETE) στο Back-End για τη διαχείριση των διαφόρων λειτουργών της εφαρμογής, όπως είναι η διαχείριση ρούχων, συνόλων, περιστάσεων και ταξιδιωτικών λιστών. Το Back-End από πλευράς του επεξεργάζεται τα αιτήματα και επικοινωνεί με τη MongoDB μέσω του Mongoose [27], για να αποθηκεύσει ή να ανακτήσει τα απαραίτητα δεδομένα. Στη συνέχεια, επιστρέφει τις πληροφορίες στο Front-End, το οποίο ενημερώνει τη διεπαφή χρήστη (UI) εξασφαλίζοντας την άμεση παρουσίαση των αλλαγών [28].

## 3.2 Αρχιτεκτονική και Σχεδιασμός

Η αρχιτεκτονική του Front-End της εφαρμογής “Digital Wardrobe” σχεδιάστηκε με στόχους την ευχρηστία, την απλότητα και την προσαρμοστικότητα. Σκοπός μας ήταν να αναπτύξουμε μια διεπαφή που να προσφέρει ομαλή πλοήγηση και θετική εμπειρία χρήστη, καθιστώντας την περιεκτική στα είδη χρηστών ανεξάρτητα από την ηλικία ή το επίπεδο εξοικείωσης/εμπειρίας με τις διαδικτυακές εφαρμογές [59]. Η υλοποίηση βασίστηκε κατά κύριο λόγο στη βιβλιοθήκη React.js, η οποία επιτρέπει τη δημιουργία μιας δυναμικής και διαδραστικής διεπαφής χρήστη [60], αλλά και πολλά ακόμη αξιοσημείωτα εργαλεία και τεχνολογίες.

### 3.2.1 Λόγος Σχεδιασμού

Ο σχεδιασμός του Front-End της εφαρμογής βασίστηκε στην ιδέα μιας φιλικής και προσιτής διεπαφής που να επιτρέπει στους χρήστες να αλληλεπιδρούν εύκολα με τις λειτουργίες της εφαρμογής. Επιλέχθηκε το React.js ώστε να διασφαλιστεί η ταχύτητα στην απόδοση της διεπαφής και η άμεση ανταπόκριση σε κάθε ενέργεια των χρηστών, ακόμη και σε

περιπτώσεις με πολλαπλές και ταυτόχρονες λειτουργίες. Επιπλέον, το React.js επιτρέπει τη δημιουργία επαναχρησιμοποιήσιμων συστατικών (components), τα οποία θα εξετάσουμε σε άλλη ενότητα, κάτι που συμβάλλει τόσο στην ευκολία αναβάθμισης της εφαρμογής όσο και στη συντήρησή της [61]. Η επιλογή του React.js, υπόλοιπων εργαλείων και τεχνολογιών καθώς και βιβλιοθηκών, θα αναλυθούν στην επόμενη ενότητα.

### 3.2.2 Ευχρηστία και Προσβασιμότητα

Η αρχιτεκτονική και ο σχεδιασμός του Front-End της εφαρμογής, βασίστηκε στην αρχή της ευχρηστίας και της προσβασιμότητας για όλους τους χρήστες, ανεξαρτήτως ηλικίας ή εμπειρίας με τη χρήση του διαδικτύου και διαδικτυακών εφαρμογών. Στόχος μας ήταν η δημιουργία μιας διεπαφής που να επιτρέπει εύκολη και γρήγορη πλοήγηση, παρέχοντας ξεκάθαρα μενού, περιεχόμενα υποσέλιδου και σαφή οργάνωση των σελίδων [59]. Η χρήση των επαναχρησιμοποιούμενων συστατικών βοήθησε στη διατήρηση καθολικής ομοιομορφίας στην εμφάνιση και λειτουργία των περιεχομένων, καθιστώντας την εφαρμογή φιλική.

Παράλληλα, δόθηκε έμφαση στη βελτιστοποίηση της προσβασιμότητας, λαμβάνοντας υπόψη τις ανάγκες από όλες τις ομάδες χρηστών και εμπειριών. Αυτό περιλαμβάνει τη χρήση ευδιάκριτων και απαλών χρωμάτων, την επιλογή αισθητικών γραμματοσειρών, τη χρήση εικονιδίων (font awesome) σε κάθε λειτουργία και την επιλογή γλαφυρών εικόνων background σε ταπετσαρία και υποσέλιδο. Η εφαρμογή σχεδιάστηκε με στόχο να παρέχει μια συνεπή και απρόσκοπη εμπειρία πλοήγησης, ενισχύοντας την αυτοπεποίθηση των χρηστών με τη χρήση διαδικτυακών εφαρμογών και διευκολύνοντας την αλληλεπίδρασή τους με αυτή [62].

## 3.3 Τεχνολογίες και Εργαλεία

Για την ανάπτυξη του Front-End της εφαρμογής “Digital Wardrobe” χρησιμοποιήθηκαν σύγχρονες τεχνολογίες και εργαλεία που συνεισφέρουν στην ευχρηστία, την προσβασιμότητα, τη διασύνδεση με το Back-End και κατ’ επέκταση τη MongoDB, αλλά και τη διευκόλυνση δημιουργίας της διεπαφής χρήστη. Κάθε τεχνολογία επιλέχθηκε με γνώμονα την αποδοτική διαχείριση του περιεχομένου, την αλληλεπίδραση με τους χρήστες και τη σταθερή και απρόσκοπη επικοινωνία με το Back-End. Στην ενότητα αυτή, θα αναλύσουμε τις γλώσσες προγραμματισμού, τα frameworks και τις βιβλιοθήκες που χρησιμοποιήθηκαν για τη συμβολή στα προαναφερθέντα. Θα παρουσιαστεί η γενική τους χρήση και θα δοθεί

βάση στους τρόπους με τους οποίους οι τεχνολογίες και τα εργαλεία αυτά επιτυγχάνουν τους στόχους της εφαρμογής μας.

### 3.3.1 HTML

Η **HTML** (HyperText Markup Language) είναι η βασική γλώσσα για τη δομή και οργάνωση του περιεχομένου μιας ιστοσελίδας. Χρησιμοποιείται για τη δημιουργία των βασικών στοιχείων που απαρτίζουν μια ιστοσελίδα, όπως επικεφαλίδες, παραγράφους, λίστες, εικόνες, και συνδέσμους. Για να προσδιοριστεί κάθε στοιχείο χρησιμοποιούνται οι ετικέτες HTML, όπως `<div>`, `<h1>`, `<p>`, `<a>` που καθορίζουν την τοποθέτηση και το ρόλο του στην ιστοσελίδα [19], [63]. Αντές παρέχουν τη βασική διάταξη του περιεχομένου και επιτρέπουν την προσθήκη άλλων γλωσσών προγραμματισμού, όπως η CSS και η JavaScript, για τη διαμόρφωση και τη λειτουργικότητα της ιστοσελίδας [12], [19]. Για τη συγγραφή κώδικα HTML με JavaScript, χρησιμοποιείται το **React JSX** (JavaScript XML), πρακτικό εργαλείο για τη δημιουργία επαναχρησιμοποιήσιμων στοιχείων UI με λιγότερες γραμμές κώδικα [64], [65].

Στην εφαρμογή, η HTML αποτελεί τη βάση για την κατασκευή όλων των σελίδων και των λειτουργιών της. Η χρήση των HTML ετικετών επιτρέπει τη δημιουργία μιας σαφούς και λειτουργικής διεπαφής χρήστη, οργανώνοντας το περιεχόμενο σε κατηγορίες όπως προφίλ χρηστών, ρούχα, σύνολα, περιστάσεις, και ταξιδιωτικές λίστες. Ο κώδικας HTML βρίσκεται ενσωματωμένος στον JavaScript κώδικα, μέσω της βιβλιοθήκης React, με στόχο τη δημιουργία δυναμικών στοιχείων, όπως μενού, φόρμες και κουμπιά [65]. Η HTML συναντάται σε όλα τα αρχεία των περιεχομένων με τα οποία αλληλεπιδρούν οι χρήστες, όπως αυτά που περιλαμβάνονται στην επικεφαλίδα και το υποσέλιδο.

Οι ετικέτες που χρησιμοποιούνται είναι οι `<h1/2/3..>` για επικεφαλίδες, `<p>` για παραγράφους, `<a href="">` για συνδέσμους, `<ul>` και `<li>` για λίστες, `<form>` για φόρμες, `<div>` για τη δομή, `<a>` και `<span>` για συνδέσμους και κείμενα, `<strong>` για έντονο κείμενο, `<button>` για την ενεργοποίηση ενεργειών, `<label>` για τον μηχανισμό μιας φόρμας, `<input>` για τα πεδία εισαγωγής δεδομένων, `<i>` για κλίση κειμένου, `<img>` για εισαγωγή εικόνων, `<select>` για μενού επιλογών και `<option>` για τις επιλογές αυτές και `<textarea>` για την εισαγωγή μεγάλων κειμένων από τους χρήστες.

### 3.3.2 CSS

Η **CSS** (Cascading Style Sheets) είναι μια γλώσσα προγραμματισμού που χρησιμοποιείται για τον σχεδιασμό και τη μορφοποίηση ιστοσελίδων και διαδικτυακών εφαρμογών. Παρέχει τη δυνατότητα προσαρμογής της εμφάνισης του περιεχομένου HTML, καθορίζοντας χρώματα, γραμματοσειρές, περιθώρια, διατάξεις και πολλά ακόμη οπτικά χαρακτηριστικά. Ο κεντρικός ρόλος της CSS είναι να διαχωρίζει το περιεχόμενο της ιστοσελίδας ή της εφαρμογής από τον σχεδιασμό της, δίνοντας τη δυνατότητα αλλαγής της εμφάνισης χωρίς να επηρεάζεται ταυτόχρονα και η δομή της. Χρησιμοποιείται επίσης για τη βελτίωση της χρηστικότητας και της εμπειρίας χρήστη. Μπορεί μια εφαρμογή να σχεδιαστεί από πολύ απλή έως και πολύ σύνθετη, με πολλά διαφορετικά animations [18], [66].

Στην εφαρμογή μας, η **CSS** χρησιμοποιείται για να καθορίσει τη μορφή και την οπτική διάταξη των στοιχείων που εμφανίζονται στο **Front-End**. Εφαρμόζονται κανόνες για να σχεδιαστούν τα κουμπιά, τα μενού, τα πλαίσια, οι φόρμες, το υποσέλιδο, οι ταπετσαρίες και όλες οι διατάξεις που αλληλεπιδρούν με τους χρήστες. Όλος ο σχεδιασμός της σελίδας προέρχεται από το αρχείο **App.css**, μέσα στο οποίο καθορίστηκαν οι βασικές οδηγίες στυλ για τη διαδικτυακή εφαρμογή, ενώ σε άλλες επιμέρους σελίδες ορίζονται ειδικές ρυθμίσεις που εξυπηρετούν διαφορετικές λειτουργίες της εφαρμογής, όπως dropdown μενού, λίστες και πλαίσια επιλογής αντικειμένων. Το App.css συνδέεται με το κύριο App.js αρχείο με την εντολή “**import './App.css';**”, καθώς βρίσκονται στον ίδιο φάκελο src. Μέσω της CSS επιτυγχάνεται η ομοιομορφία στην εμφάνιση και η αισθητική της εφαρμογής, πετυχαίνοντας τους στόχους της και παρουσιάζοντας στους χρήστες μια απλά και λιτά σχεδιασμένη εφαρμογή με συνδυασμό άνετων και χαλαρωτικών χρωμάτων.

### 3.3.3 JavaScript

Η **JavaScript** είναι μια δημοφιλής γλώσσα προγραμματισμού που χρησιμοποιείται για τη δημιουργία διαδραστικών στοιχείων σε ιστοσελίδες και εφαρμογές. Συνδυάζεται επίσης με την HTML και την CSS για να επιτυγχάνει δυναμικές λειτουργίες όπως η επεξεργασία μιας φόρμας, η δημιουργία διαφόρων animation και η επικοινωνία με εξωτερικούς διακομιστές (Back-End & MongoDB) μέσω HTTP αιτημάτων [22], . Παρέχει τη δυνατότητα αλληλεπίδρασης των χρηστών με τα περιεχόμενα της ιστοσελίδας και τη διαχείριση του περιβάλλοντος χρήστη (UI). Η JavaScript χρησιμοποιείται ευρέως για την ενίσχυση της εμπειρίας χρήστης μέσω δυναμικών περιεχομένων και λειτουργιών [66].

Στην εφαρμογή μας, η **JavaScript** χρησιμοποιείται για να υλοποιεί τις διαδραστικές λειτουργίες και να διαχειρίζεται τα δεδομένα που εισάγονται από τους χρήστες σε κάθε περίπτωση. Αντίστοιχα με την HTML, χρησιμοποιείται σε όλα τα αρχεία της εφαρμογής μας,

όπως το βασικό App.js, και σε κάθε συστατικό του components, όπως το Clothes.js, Outfits.js, MyAccount.js, Footer.js και πολλά ακόμη αρχεία. Η JavaScript, μέσω του React.js που θα αναλύσουμε στην επόμενη υπο-ενότητα, επιτρέπει την επικοινωνία με το Back-End και κατ' επέκταση τη βάση δεδομένων, διαχειρίζεται την απόδοση της εφαρμογής και εξασφαλίζει την ομαλή και ασφαλή αλληλεπίδραση των χρηστών με τη διεπαφή [28].

### 3.3.4 React.js

To **React.js** είναι μια βιβλιοθήκη JavaScript που αναπτύχθηκε από το Facebook για τη δημιουργία σύγχρονων διαδραστικών και δυναμικών διεπαφών χρήστη (UI). Χρησιμοποιείται ευρέως λόγω των πλεονεκτημάτων του, όπως η ταχύτητα ανάπτυξης, η δυνατότητα επαναχρησιμοποίησης συστατικών (components), επιτρέποντας στους προγραμματιστές να κατασκευάζουν κώδικα με αρθρωτό και δομημένο τρόπο, και η απόδοση ενημερώσεων χάρη στον Virtual DOM [65]. Ενσωματώνεται με άλλες βιβλιοθήκες και frameworks και υποστηρίζει την κατασκευή μονοσέλιδων εφαρμογών (SPAs) [68]. Εναλλακτικές του React.js περιλαμβάνουν τα Vue.js, Angular, και Svelte, αλλά αυτό ξεχωρίζει για την ευελιξία του και την κοινότητα υποστήριξης του.

Στην εφαρμογή “Digital Wardrobe”, το **React.js** είναι η κύρια σε χρήση τεχνολογία για την κατασκευή όλων των δυναμικών συστατικών του Front-End. Με αυτό, δημιουργήθηκαν οι βασικές σελίδες και διαδραστικές λειτουργίες της εφαρμογής, όπως οι φόρμες, η πλοήγηση μεταξύ των σελίδων και η διαχείριση δεδομένων, ενώ παράλληλα εξασφαλίζεται η απρόσκοπη επικοινωνία με το Back-End μέσω του Axios και άλλων βιβλιοθηκών. Γίνεται χρήση της σε όλα τα αρχεία συστατικών του φακέλου components, όπως τα Header.js, EditCloth.js, CreateOutfit.js, και άλλα. Για την εγκατάστασή του χρησιμοποιήθηκε στη γραμμή εντολών η εντολή “**npm install -g create-react-app**”, για την δημιουργία του project της εφαρμογής η “**npx create-react-app digital\_wardrobe**” και για την εκκίνηση της εφαρμογής η “**npm start**” εντός του φακέλου src [16].

### 3.3.5 Axios

To **Axios** είναι μια δημοφιλής βιβλιοθήκη JavaScript η οποία χρησιμοποιείται για την αποστολή αιτημάτων HTTP σε APIs από το Front-End, διευκολύνοντας την επικοινωνία μεταξύ των τμημάτων της εφαρμογής. Υποστηρίζει αιτήματα με τις γνωστές μεθόδους GET, POST, PUT και DELETE, και επιτρέπει τη διαχείριση των promises, διευκολύνοντας τον ασύγχρονο προγραμματισμό [58]. Η βιβλιοθήκη προσφέρει επίσης αυτόματη μετατροπή

δεδομένων σε JSON, υποστήριξη CORS, που αναλύσαμε στο προηγούμενο κεφάλαιο, για αιτήματα διασύνδεσης διαφορετικών domains και διευκολύνει τον χειρισμό σφαλμάτων [45]. Η εγκατάστασή της πραγματοποιείται στη γραμμή εντολών με την εντολή “**npm install axios**” και εισάγεται στα αρχεία όπου χρειάζεται με την εντολή “**import axios from 'axios';**” [69].

Στην εφαρμογή, το **Axios** σε συνδυασμό με τις λέξεις-κλειδιά **async/await** χρησιμοποιείται για την αποστολή αιτημάτων HTTP στο Back-End και την ανάκτηση ή αποστολή δεδομένων από τη βάση δεδομένων MongoDB. Τα **async/await** επιτρέπουν την αναμονή για την ολοκλήρωση ενός αιτήματος πριν από την συνέχεια εκτέλεσης του κώδικα, χωρίς όμως να επηρεάζεται η διεπαφή χρήστη. Με τον συνδυασμό αυτόν, επιτυγχάνεται καλύτερη διαχείριση των αιτημάτων και απλούστερη σύνταξη κώδικα, ειδικά όταν απαιτούνται πολλαπλές κλήσεις σε APIs [55]. Το Axios χρησιμοποιείται σε αρχεία που απαιτούν την επικοινωνία του Front-End με το Back-End, διασφαλίζοντας την αποστολή των δεδομένων που εισάγουν οι χρήστες και την ανάκτηση των απαραίτητων πληροφοριών [58]. Αυτά τα αρχεία αποτελούν τα Clothes.js, Outfit.js, Occasion.js, TravelLists.js, MyAccount.js και τα επιμέρους Edit & Create αρχεία τους, καθώς και τα Signin.js και Signup.js.

Στο παρακάτω παράδειγμα, διακρίνεται η χρήση του Axios σε συνδυασμό με τα **async/await**. Στο τμήμα κώδικα πραγματοποιείται η σύνδεση χρηστών με την χρήση της συνάρτησης “**handleSubmit**”, η οποία εκτελεί ένα POST αίτημα με τη συνοδεία των email και κωδικού πρόσβασης που εισάγουν οι χρήστες. Εφόσον το αίτημα είναι επιτυχές (έγκυρα στοιχεία), αποθηκεύεται το access token και οι χρήστες οδηγούνται στην αρχική σελίδα με όλες τις λειτουργίες, αλλιώς εμφανίζεται το μήνυμα “**Invalid Credentials**” (Εικόνα 15).

```
const handleSubmit = async (e) => {
  e.preventDefault();
  try {
    const response = await axios.post('http://localhost:3000/api/auth/signin', { email, password });
    const { accessToken } = response.data;
    console.log('Access Token:', response.data.accessToken);
    onSignIn(accessToken);
    localStorage.setItem('token', accessToken);
    navigate('/');
  } catch (error) {
    setError('Invalid credentials');
  }
};
```

**Εικόνα 15:** Αποστολή αιτήματος POST για την σύνδεση του χρήστη στη διαδικτυακή εφαρμογή (τμήμα κώδικα από το αρχείο Signin.js).

Στην παρακάτω εικόνα, παρατηρείται η χρήση του Axios σε συνδυασμό με τα **async/await** για την ανάκτηση και προβολή των ρούχων των χρηστών. Συγκεκριμένα, με την συνάρτηση

“fetchClothes” η οποία εκτελείται ασύγχρονα (async), λαμβάνει το token που αποθηκεύτηκε και το χρησιμοποιεί για την αποστολής αιτήματος GET χρησιμοποιώντας την εντολή await. Εάν το αίτημα είναι επιτυχές, τα δεδομένα αποθηκεύονται στη μεταβλητή state “setClothes”.

```
const fetchClothes = async () => {
  try {
    const token = localStorage.getItem('token');
    const config = {
      headers: { Authorization: `Bearer ${token}` },
      withCredentials: true
    };
    const response = await axios.get('http://localhost:3000/api/clothes', config);
    setClothes(response.data);
  } catch (error) {
    console.error('Error fetching clothes:', error.response ? error.response.data : error.message);
  }
};
```

**Εικόνα 16:** Αποστολή αιτήματος GET για την ανάκτηση όλων των ρούχων του χρήστη και την προβολή τους στη σελίδα (τμήμα κώδικα από το αρχείο Clothes.js).

## 3.4 Υλοποίηση Διεπαφής και Εμπειρίας Χρήστη

Σε αυτή την ενότητα θα εξετάσουμε βήμα-βήμα τις ενέργειες που πραγματοποιήθηκαν για τη δημιουργία και τον σχεδιασμό του Front-End της εφαρμογής “Digital Wardrobe”. Θα παρουσιαστούν οι αρχικές ρυθμίσεις που έγιναν για την οργάνωση του περιβάλλοντος και οι διαδικασίες ανάπτυξης όλων των λειτουργιών, όπως τα Header και Footer, οι φόρμες εισόδου και εγγραφής, τα περιεχόμενα κάθε βασικής λειτουργίας της εφαρμογής, καθώς και οι οπτικές και λειτουργικές βελτιώσεις που επιλέχθηκαν για να ενισχύσουν την διεπαφή και την εμπειρία χρήστης.

### 3.4.1 Αρχικός Σχεδιασμός και Στήσιμο της Εφαρμογής

Αφού εγκαταστάθηκε το React.js και δημιουργήθηκε το project του “Digital Wardrobe” με τις κατάλληλες εντολές στη γραμμή εντολών (CMD), μέσα στον φάκελο src δημιουργήσαμε έναν νέο φάκελο με όνομα components. Σε αυτό το φάκελο βρίσκονται όλες οι λειτουργίες της εφαρμογής σε ξεχωριστά JSON αρχεία, όπως η επικεφαλίδα, το υποσέλιδο, τα ρούχα, τα σύνολα, οι περιστάσεις, οι ταξιδιωτικές λίστες και το προφίλ χρήστη. Καθένα από αυτά συμπεριλαμβάνεται στο βασικό αρχείο App.js με import. Για να επιτευχθεί η πλοήγηση μεταξύ των σελίδων, εγκαταστάθηκε το React Router με την εντολή “npm install react-router-dom” και συμπεριλαμβάνεται στο App.js με την εντολή “import { BrowserRouter as Router, Route, Routes, Link } from 'react-router-dom';”. Στη

συνέχεια, η εφαρμογή εκτελέστηκε τοπικά με την εντολή “**npm start**”, εμφανίζοντας την αρχική της μορφή σε μια απλή σελίδα στο πρόγραμμα περιήγησης. Από αυτό το σημείο και ύστερα πραγματοποιήθηκαν μεθοδικά οι προσθήκες και αλλαγές [65], [71]. Στη συνέχεια, θα αναλύσουμε κάθε στοιχείο της διαδικτυακής εφαρμογής, τις υπηρεσίες που προσφέρονται και την ιδιαίτερη σχεδιαστική προσέγγιση.

### 3.4.2 Σύνδεση, Εγγραφή και Προφίλ Χρηστών

Κάποιες από τις πρώτες λειτουργίες που υλοποιήσαμε ήταν σύνδεση/αποσύνδεση και εγγραφή των χρηστών, αντικείμενα που δημιουργήθηκαν παράλληλα με το αρχείο “auth\_routes.js” του Back-End. Αυτές οι λειτουργίες αποτελούν θεμέλιο για την εξατομικευμένη εμπειρία στην εφαρμογή μας, καθώς σε αντίθεση με τους μη εγγεγραμμένους χρήστες, οι εγγεγραμμένοι έχουν πρόσβαση σε όλες τις βασικές λειτουργίες. Για την **εγγραφή** τους (Sign Up) δημιουργήθηκε μια φόρμα συμπλήρωσης στοιχείων, στην οποία καλούνται να προσθέσουν απαίτουμενα στοιχεία, τα οποία συνοδεύονται από αστερίσκο (\*), όπως e-mail, κωδικό πρόσβασης, όνομα, επώνυμο και εικόνα προφίλ, καθώς και προαιρετικά ηλικία και αγαπημένα χρώματα. Για την **είσοδο** στην εφαρμογή (Sign In), δημιουργήθηκε μια φόρμα σύνδεσης, όπου οι χρήστες εισάγουν το email και τον κωδικό πρόσβασης που χρησιμοποίησαν για την εγγραφή. Αναπτύξαμε και την επιλογή **αποσύνδεσης** (Sign Out). Η χρήση στοιχείων ασφαλείας, όπως η αποθήκευση tokens για την ταυτοποίηση και ο έλεγχος πρόσβασης με ασφαλή headers, εξασφαλίζει την προστασία των δεδομένων.

The image consists of two side-by-side screenshots of a mobile application. The left screenshot shows the 'Sign Up' registration screen. It features a title 'Sign Up' at the top. Below it are seven input fields with labels: 'E-mail\*', 'Password\*', 'Name\*', 'Surname\*', 'Age', 'Favorite Colors', and 'Profile Picture'. Each field has a placeholder text inside. A large orange 'Sign Up' button is at the bottom. The right screenshot shows a 'Welcome to Digital Wardrobe!' screen. It has a brown background with white text. At the top is the title 'Welcome to Digital Wardrobe!'. Below it is a paragraph: 'Join us and manage your wardrobe digitally and efficiently today!'. Underneath that is another paragraph: 'If you wish to learn more about us before signing up, please visit our [About Us](#) page.' Both screenshots have a blurred background image of what appears to be a clothing store interior.

**Εικόνα 17:** Φόρμα εγγραφής χρηστών στο Digital Wardrobe, με μήνυμα καλωσορίσματος και καθοδήγηση στο About Us.

Με την ολοκλήρωση της εισόδου μπορούν να επισκεφτούν το προφίλ τους από την επικεφαλίδα και συγκεκριμένα επιλέγοντας το “**My Account**”. Εκεί μπορούν να προβάλλουν την καρτέλα τους με όλες τις λεπτομέρειες που έχουν συμπληρώσει αλλά και να τις επεξεργαστούν ή να διαγράψουν το προφίλ τους. Κατά την επεξεργασία, τα προηγούμενα στοιχεία είναι ήδη συμπληρωμένα με τις πληροφορίες που είχαν εισάγει. Με κάθε αλλαγή ενημερώνεται κατάλληλα το Back-End με τα νέα δεδομένα που εισάγονται [28].

<b>Current Profile Picture</b>	
<b>Επιλογή αρχείου</b> Δεν επιλέχθηκε κανένα αρχείο.	
<b>Name:</b> <input type="text" value="Tony"/>	
<b>Surname:</b> <input type="text" value="Stark"/>	
<b>Email:</b> <input type="text" value="tonystark@yahoo.gr"/>	
<b>Password:</b> <input type="text" value="New password"/>	
<b>Age:</b> <input type="text" value="53"/>	
<b>Favorite Color\`s:</b> <input type="text" value="Red, Gold"/>	
<input type="button" value="Update Profile"/> <input type="button" value="Cancel"/>	

**Εικόνα 18:** Καρτέλα επεξεργασίας προφίλ με προσυμπληρωμένα στοιχεία χρήστη και πλήκτρα “Update Profile” και “Cancel”.

### 3.4.3 Αρχική, Επικεφαλίδα & Υποσέλιδο: Δομή και Λειτουργίες

Η αρχική σελίδα (**Home**) της εφαρμογής μας αποτελεί τον πρώτο χώρο επαφής των χρηστών με το “Digital Wardrobe”. Σχεδιάστηκε με προσοχή ώστε να προσφέρει μια φιλόξενη και ευχάριστη εμπειρία κατά την χρήση του. Κεντρικό στοιχείο είναι η ταπετσαρία με θέμα ένδυσης, με γυναικεία και ανδρικά ρούχα, η οποία ενισχύει τον χαρακτήρα της εφαρμογής. Στο κέντρο της αρχικής σελίδας εμφανίζεται το μήνυμα καλωσορίσματος “Welcome to your Digital Wardrobe” παρέχοντας από κάτω στους χρήστες μια ιδιαίτερη περιγραφή του τι θα

βρουν στην εφαρμογή. Αριστερά από το μήνυμα υποδοχής, εμφανίζεται η ενότητα “News & Updates” η οποία περιέχει ενημερώσεις και νέες δυνατότητες που προστέθηκαν στην εφαρμογή ώστε οι χρήστες να γνωρίζουν κάθε προσθήκη [62].

Η επικεφαλίδα (**Header**) της εφαρμογής περιλαμβάνει όλες τις βασικές λειτουργίες και είναι δυναμική, αναπροσαρμόζοντας το περιεχόμενό της ανάλογα με το αν οι χρήστες είναι συνδεδεμένοι ή όχι. Όταν είναι, η επικεφαλίδα περιλαμβάνει επιλογές όπως “Home”, “Clothes”, “Outfits”, “Occasions” και “Travel Lists”, καθώς και τα “My Account” και “Sign Out”. Αν δεν είναι, εμφανίζονται μόνο οι επιλογές “Home”, “Sign In” και “Sign Up”. Παράλληλα, στην αριστερή της πλευρά βρίσκεται το λογότυπο της εφαρμογής και η ονομασία γραμμένη σε καλλιγραφική μορφή. Το background της επικεφαλίδας απεικονίζει ένα κοντινό πλάνο μασίφ ξύλου, υλικό που χρησιμοποιείται στις περισσότερες ντουλάπες. Τόσο η επικεφαλίδα όσο και η αρχική της εφαρμογής παρατηρούνται στην παρακάτω εικόνα.



**Εικόνα 19:** Η οπτική ενός συνδεδεμένου χρήστη από την αρχική του “Digital Wardrobe” μαζί με την επικεφαλίδα.

Το υποσέλιδο (**Footer**) της σελίδας δομείται από διάφορες κατηγοριοποιημένες επιλογές που επιτρέπουν στους χρήστες να εξερευνήσουν και να ανακαλύψουν περισσότερες πληροφορίες σχετικά με την εφαρμογή και τις υπηρεσίες που προσφέρει. Συγκεκριμένα, περιλαμβάνει την ενότητα με όνομα “Digital Wardrobe” με επιλογές όπως “About Us” στην οποία μπορούν να γνωρίσουν την εφαρμογή και τους στόχους της, “Our Services” που παρουσιάζονται οι υπηρεσίες της εφαρμογής, “Privacy Policy” με την πολιτική απορρήτου, “Terms & Conditions” με τους όρους και προϋποθέσεις χρήσης, και “User Agreement” με τους όρους που έχουν αποδεχτεί οι χρήστες. Υπάρχει επίσης η ενότητα “Get Help” με επιλογές όπως “How It Works” που περιλαμβάνει τον οδηγό χρήσης, “Help Center” που περιλαμβάνει μια

φόρμα επικοινωνίας για τυχόν προβλήματα ή προτάσεις, “FAQ” που περιλαμβάνει μια λίστα από dropdown ερωτήσεις και απαντήσεις, και “Contact Us” με όλους τους τρόπους επικοινωνίας με την ομάδα μας.

Επιπρόσθετα, το υποσέλιδο προσφέρει τη δυνατότητα εγγραφής στο **newsletter**, με την εισαγωγή του email, μέσω της ενότητας “Stay Updated” για την λήψη των τελευταίων ενημερώσεων και ενισχύσεων στην εφαρμογή. Τέλος, στην ενότητα “Follow Us On.” παρέχονται τα animated εικονίδια κοινωνικών δικτύων για εύκολη πρόσβαση στους λογαριασμούς κοινωνικής δικτύωσης της εφαρμογής.



**Εικόνα 20:** Υποσέλιδο της εφαρμογής με όλες τις ενότητες και τις διαθέσιμες επιλογές τους.

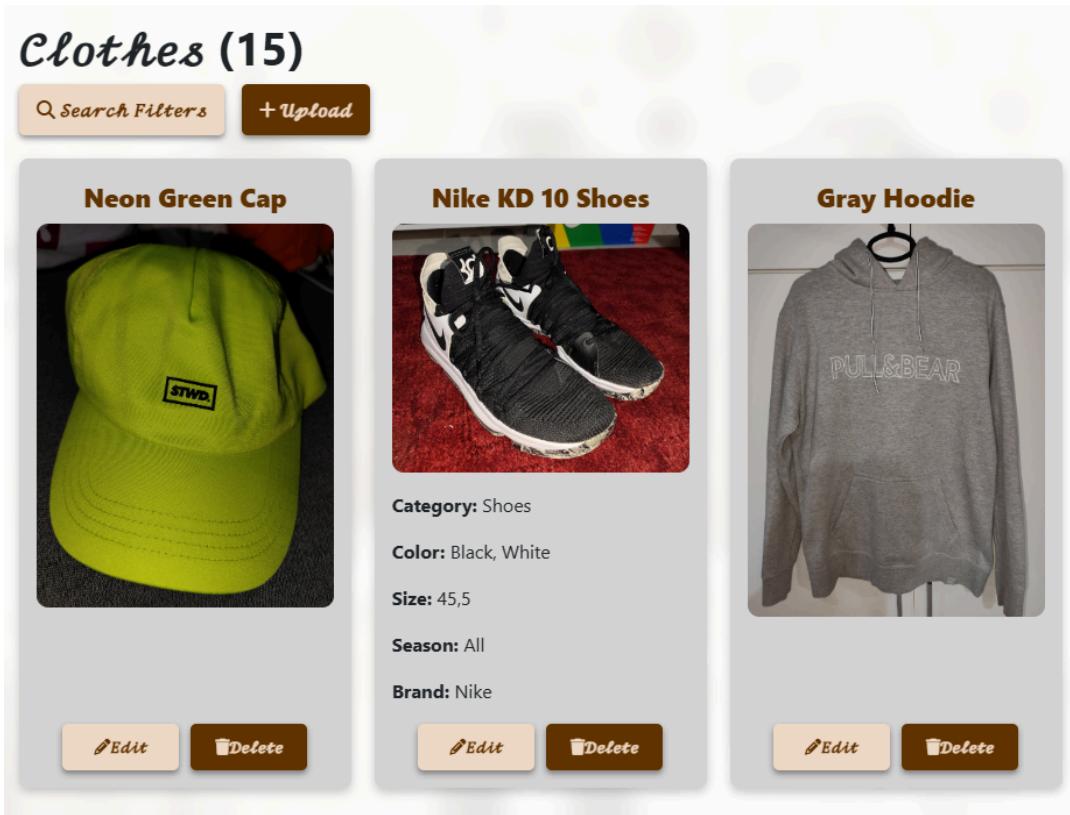
### 3.4.4 Βασικές Λειτουργίες

Με την ολοκλήρωση της υλοποίησης της σύνδεσης/αποσύνδεσης και εγγραφής χρηστών, αρχικής της εφαρμογής, επικεφαλίδας με τις βασικές λειτουργίες, και υποσέλιδου με τις τέσσερις βασικές ενότητες και τις παρεχόμενες επιλογές, ακολούθησε η υλοποίηση των τεσσάρων βασικών λειτουργιών. Αυτές τις λειτουργίες αποτελούν τα Ρούχα, Σύνολα, Περιστάσεις και Ταξιδιωτικές Λίστες, οι οποίες βασίζονται σε παρόμοιες μεθόδους προσθήκης, επεξεργασίας, ανάκτησης και διαγραφής δεδομένων. Για κάθε ενέργεια, το Front-End επικοινωνεί με το Back-End, ενημερώνεται η βάση δεδομένων και ολοκληρώνονται τα αιτήματα με τις κατάλληλες απαντήσεις [28].

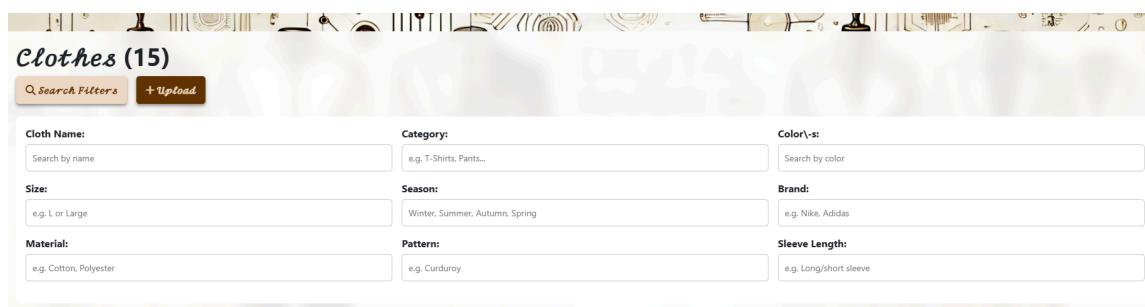
#### 3.4.4.1 Ρούχα

Στην καρτέλα “**Clothes**” οι χρήστες έχουν τη δυνατότητα να προβάλουν και να διαχειρίζονται όλα τα είδη ένδυσης που έχουν προσθέσει στο προφίλ τους. Στη σελίδα παρουσιάζεται η λίστα με τα ανεβασμένα ρούχα, όπου για κάθε ένα από αυτά ορατά είναι το όνομα και η εικόνα του. Για την εμφάνιση περισσότερων συμπληρωμένων πληροφοριών σχετικά με το

ρούχο, οι χρήστες πρέπει να κάνουν κλικ πάνω στο ρούχο το οποίο τους ενδιαφέρει, κάτι που ενεργοποιεί την λειτουργία dropdown με αυτές. Στην κορυφή της σελίδας υπάρχουν δύο βασικές επιλογές: το κουμπί “**Search Filters**” το οποίο με κλικ κάνει επίσης dropdown τα φίλτρα αναζήτησης τα οποία αποτελούνται από κάθε πεδίο προς συμπλήρωση κατά το ανέβασμα ενός ρούχου, και το κουμπί “**Upload**” το οποίο ανακατευθύνει τους χρήστες στη φόρμα προσθήκης ενός νέου ρούχου στο προφίλ τους.



**Εικόνα 21:** Τρόπος εμφάνισης των ανεβασμένων ρούχων ενός χρήστη στο προφίλ του. Το ενδιάμεσο ρούχο έπειτα από κλικ εμφανίζει περισσότερες λεπτομέρειες.



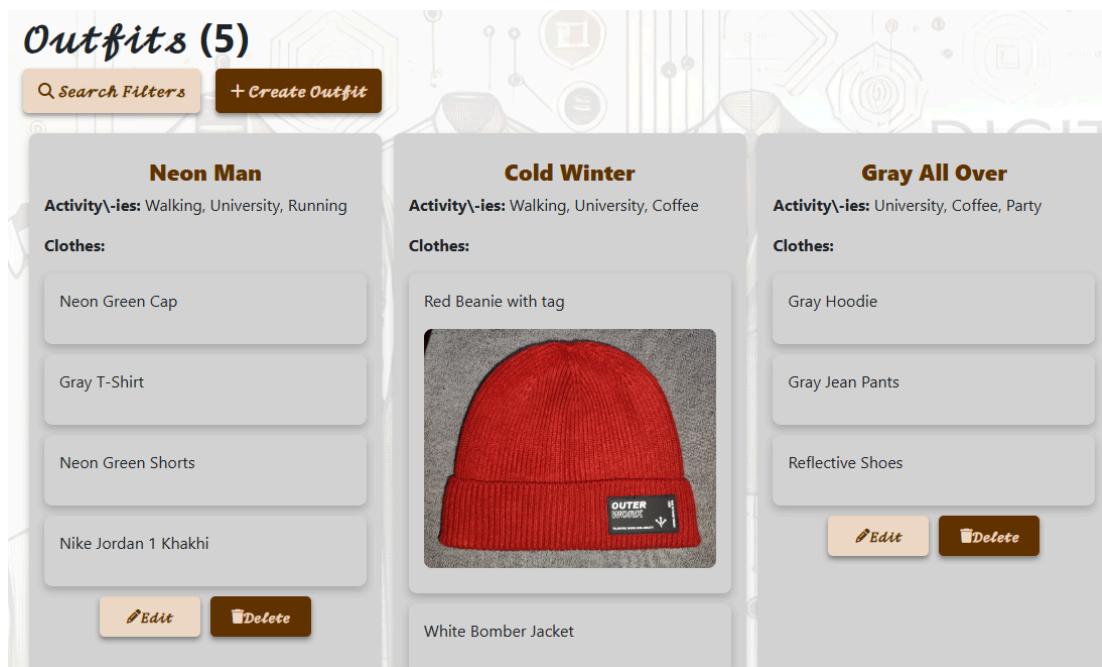
**Εικόνα 22:** Dropdown των φίλτρων αναζήτησης των ρούχων με το πάτημα του κουμπιού “Search Filters”.

Για κάθε ρούχο που έχει ανέβει στο προφίλ, υπάρχουν επιλογές “**Edit**” και “**Delete**”. Το κουμπί της επεξεργασίας ανακατευθύνει τους χρήστες σε μια σελίδα φόρμας με όνομα “**Edit Cloth**” όπου μπορούν να επεξεργαστούν το ρούχο με τα προηγούμενα στοιχεία να είναι προ-συμπληρωμένα προσφέροντας ευκολία στην ενημέρωση. Στην περίπτωση που κάνουν κλικ στο κουμπί της διαγραφής, το ρούχο διαγράφεται άμεσα από το προφίλ. Σε κάθε ενέργεια η βάση δεδομένων ενημερώνεται ανάλογα. Σε μια παρένθεση δίπλα από τον τίτλο “Clothes” εμφανίζεται ο συνολικός αριθμός των ανεβασμένων ρούχων, ο οποίος ενημερώνεται αυτόματα με κάθε προσθήκη ή αφαίρεση μέσω ενός counter.

Η επιλογή **Upload** ανακατευθύνει τους χρήστες στη φόρμα “**Upload Clothes**” όπου επιτρέπεται στους χρήστες να προσθέσουν ένα νέο ρούχο, συμπληρώνοντας απαιτούμενα πεδία, όπως το όνομα, την κατηγορία, το χρώμα, την εποχή χρήσης του και μια εικόνα, ενώ μπορεί να αφήσει κενά όσα πεδία δεν είναι υποχρεωτικά όπως το μέγεθος, μάρκα, υλικό υφάσματος, σχέδιο και μήκος μανικιών.

#### 3.4.4.2 Σύνολα

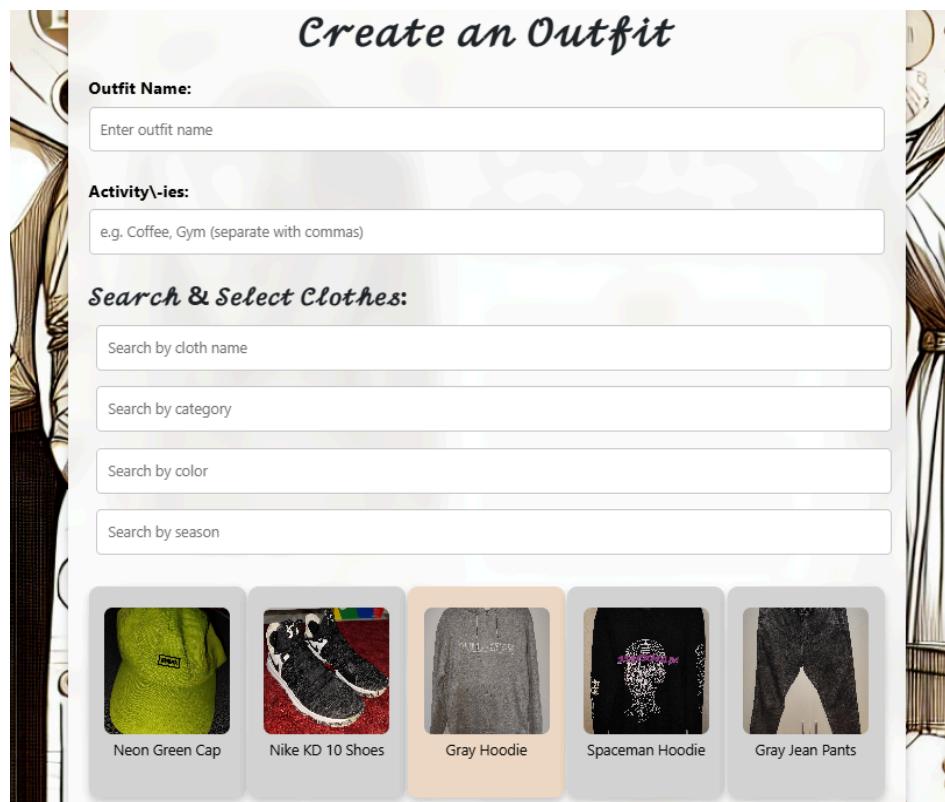
Στην καρτέλα “**Outfits**” οι χρήστες έχουν την δυνατότητα να προβάλουν και να διαχειρίζονται όλα τα σύνολα που έχουν δημιουργήσει στο προφίλ τους. Στη σελίδα παρουσιάζονται σε σειρά το ένα δίπλα στο άλλο τα δημιουργημένα σύνολα όπου για κάθε ένα από αυτά είναι ορατή η ονομασία τους, οι δραστηριότητες στις οποίες αυτά φοριούνται καθώς και τα ρούχα από τα οποία αποτελούνται. Τα ρούχα που περιέχει κάθε σύνολο παρουσιάζονται μόνο με την ονομασία τους, ωστόσο εάν οι χρήστες έχουν ξεχάσει σε ποιο ρούχο αντιστοιχεί ο τίτλος, μπορούν να κάνουν κλικ πάνω σε αυτό και με λειτουργία dropdown εμφανίζεται η εικόνα του. Στην κορυφή της σελίδας υπάρχουν δύο βασικές επιλογές: το κουμπί “**Search Filters**” που έχει παρόμοια λειτουργία με αυτό των ρούχων και το κουμπί “**Create Outfit**” το οποίο ανακατευθύνει τους χρήστες στη φόρμα δημιουργίας ενός συνόλου.



**Εικόνα 23:** Τρόπος εμφάνισης των δημιουργημένων συνόλων ενός χρήστη. Στο ενδιάμεσο σύνολο έπειτα από κλικ στον τίτλο “Cold Winter” εμφανίζεται η εικόνα του ρούχου.

Αντίστοιχα με τα ρούχα, κάθε σύνολο έχει τις επιλογές “Edit” και “Delete”. Το κουμπί της επεξεργασίας τους ανακατευθύνει σε μια σελίδα φόρμας με όνομα “Edit Outfit” όπου μπορούν να επεξεργαστούν το σύνολο με τα προηγούμενα στοιχεία να είναι προ-συμπληρωμένα και τα ρούχα να είναι ήδη επιλεγμένα. Με το κουμπί διαγραφής, το σύνολο αφαιρείται απευθείας χωρίς όμως να διαγράφεται και το ρούχο από το προφίλ. Στην παρένθεση δεξιά του τίτλου “Outfits” εμφανίζονται τα συνολικά δημιουργημένα σύνολα.

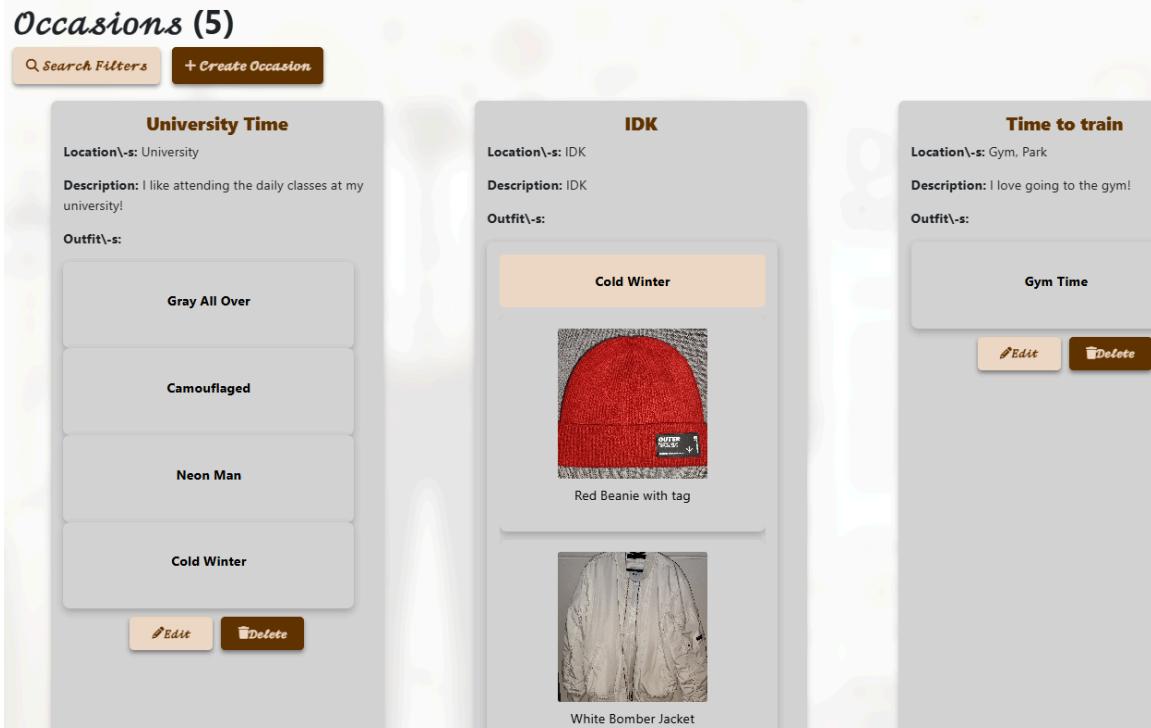
Η επιλογή **Create Outfit** ανακατευθύνει τους χρήστες στη φόρμα **“Create an Outfit”** όπου μπορούν να δημιουργήσουν ένα νέο σύνολο, συμπληρώνοντας το πεδίο του ονόματος και των δραστηριοτήτων και επιλέγοντας τα ρούχα που θα το αποτελούν. Για την εύρεση ενός είδους ένδυσης προστέθηκε μια μηχανή αναζήτησης με τα τέσσερα βασικά πεδία (όνομα ρούχου, κατηγορία, χρώμα και εποχή) (ίδια λειτουργία και στην επεξεργασία). Κάθε ρούχο στο πεδίο επιλογής παρουσιάζεται από την εικόνα του και το όνομά του μέσα σε ένα γκρι πλαίσιο. Όταν οι χρήστες κάνουν κλικ πάνω του, από γκρι μετατρέπεται σε μπεζ ώστε να δοθεί σήμανση ότι έχει επιλεχθεί.



**Εικόνα 24:** Φόρμα δημιουργίας συνόλου με τα πεδία συμπλήρωσης, τα φίλτρα αναζήτησης ρούχου και ένα μαρκαρισμένο επιλεγμένο ρούχο (Gray Hoodie).

### 3.4.4.3 Περιστάσεις

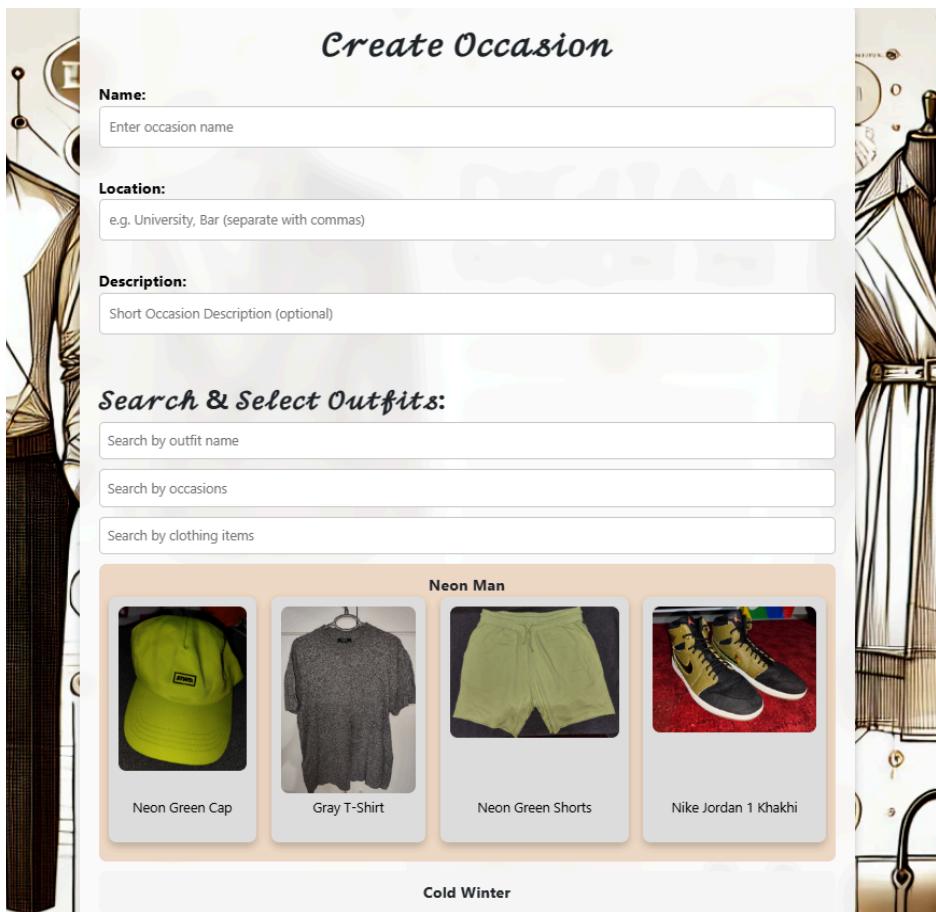
Στην καρτέλα “**Occasions**” οι χρήστες έχουν τη δυνατότητα να προβάλουν και να διαχειρίζονται όλες τις περιστάσεις που έχουν δημιουργήσει στο προφίλ τους. Όπως και στις προηγούμενες δύο σελίδες, οι περιστάσεις εμφανίζονται η μια δίπλα στην άλλη όπου για κάθε μια από αυτές είναι ορατή η ονομασία, η τοποθεσία, μια περιγραφή και η λίστα με τα περιλαμβανόμενα σύνολα. Τα σύνολα εμφανίζονται μόνο με το όνομά τους, ωστόσο εάν οι χρήστες κάνουν κλικ πάνω σε αυτά έχει δημιουργηθεί παρόμοιο dropdown με αυτό των “Outfits” που εμφανίζει τα περιεχόμενα είδη ένδυσης με εικόνα και όνομα. Στην κορυφή της σελίδας υπάρχουν δύο βασικές επιλογές: το κουμπί “**Search Filters**” που έχει παρόμοια λειτουργία με αυτό των ρούχων και των συνόλων, και το κουμπί “**Create Occasion**” το οποίο ανακατευθύνει τους χρήστες στη φόρμα δημιουργίας μιας περίστασης.



**Εικόνα 25:** Τρόπος εμφάνισης των περιστάσεων ενός χρήστη. Στην ενδιάμεση περίσταση έπειτα από κλικ στο σύνολο με όνομα “Cold Winter” εμφανίζονται τα περιεχόμενα ρούχα.

Όπως τα ρούχα και τα σύνολα, έτσι και κάθε περίσταση έχει τις επιλογές “Edit” και “Delete”. Το κουμπί της επεξεργασίας ανακατευθύνει σε μια σελίδα φόρμας με όνομα “Edit Occasion” όπου μπορούν να επεξεργαστούν την περίσταση με όλα τα προηγούμενα στοιχεία προ-συμπληρωμένα και τα σύνολα ήδη επιλεγμένα. Με το κουμπί της διαγραφής, η περίσταση αφαιρείται απευθείας χωρίς όμως να επηρεάζονται από αυτό τα περιεχόμενα σύνολα και τα περιεχόμενα είδη ένδυσης. Στην παρένθεση δεξιά του “Occasions” παρατηρείται ο συνολικός αριθμός δημιουργημένων περιστάσεων.

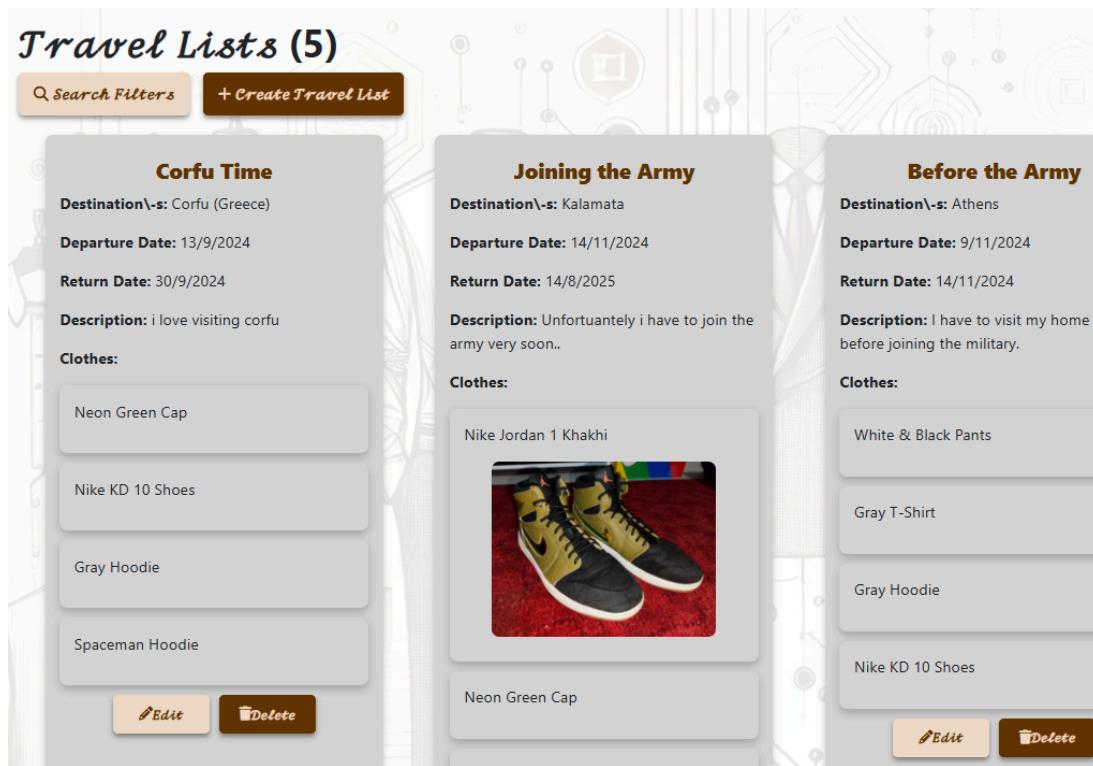
Η επιλογή Create Occasion οδηγεί στη φόρμα “Create Occasion” όπου μπορούν οι χρήστες να δημιουργήσουν μια νέα περίσταση συμπληρώνοντας στοιχεία όπως ονομασία, τοποθεσία, περιγραφή και επιλέγοντας τα κατάλληλα σύνολα από την λίστα επιλογής. Για την εύρεση ενός συνόλου έχει προστεθεί μια μηχανή αναζήτησης με βάση την ονομασία, τις περιστάσεις χρήσης και τα περιεχόμενα ρούχα (όνομα ρούχου, κατηγορία, χρώμα και εποχή) (ίδια λειτουργία και στην επεξεργασία). Κάθε σύνολο στη λίστα επιλογής παρουσιάζεται με την ονομασία του σε ένα γκρι πλαίσιο, το οποίο μετατρέπεται σε μπεζ όταν επιλεγεί, εμφανίζοντας παράλληλα τα περιεχόμενα ρούχα σε εικόνες.



**Εικόνα 26:** Φόρμα δημιουργίας περίστασης με τα πεδία συμπλήρωσης, τα φίλτρα αναζήτησης συνόλου και ένα μαρκαρισμένο επιλεγμένο σύνολο (*Neon Man*).

#### 3.4.4.4 Ταξιδιωτικές Λίστες

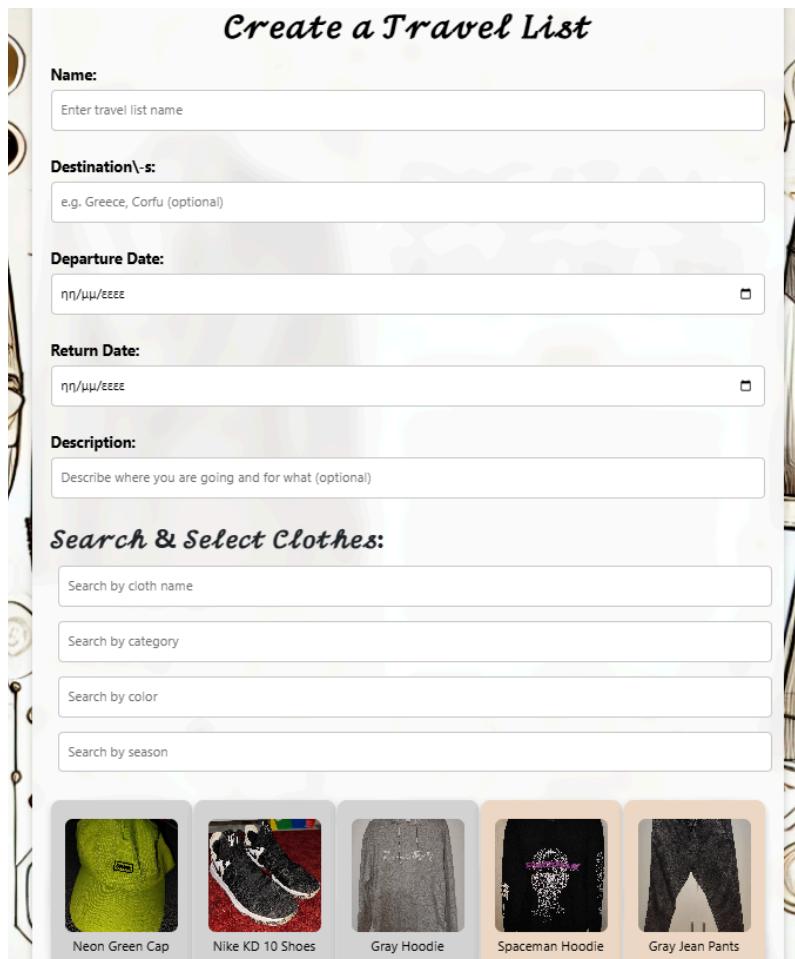
Στην καρτέλα “Travel Lists” οι χρήστες μπορούν να προβάλλουν τις ταξιδιωτικές λίστες που έχουν δημιουργήσει και να τις διαχειριστούν. Όπως και στις προηγούμενες τρεις σελίδες, οι ταξιδιωτικές λίστες εμφανίζονται η μια δίπλα στην άλλη όπου για κάθε μια από αυτές είναι ορατή η ονομασία, ο προορισμός, οι ημερομηνίες αναχώρησης και επιστροφής, μια περιγραφή και τα ρούχα που περιλαμβάνονται. Τα περιεχόμενα ρούχα όπως και στα σύνολα, εμφανίζονται μόνο με το όνομά τους, αλλά με την λειτουργία dropdown που έχει επίσης ενσωματωθεί, με ένα κλικ εμφανίζεται η εικόνα του αντικειμένου. Στην κορυφή της σελίδας υπάρχουν δύο βασικές επιλογές: το κουμπί “Search Filters” που έχει παρόμοια λειτουργία με αυτό των ρούχων, συνόλων και περιστάσεων, και το κουμπί “Create Travel List” το οποίο ανακατευθύνει τους χρήστες στη φόρμα δημιουργίας μιας ταξιδιωτικής λίστας.



**Εικόνα 27:** Τρόπος εμφάνισης των ταξιδιωτικών λιστών ενός χρήστη. Στην ενδιάμεση ταξιδιωτική λίστα έπειτα από κλικ στο ρούχο με όνομα “Nike Jordan 1 Khakhi” εμφανίζεται η εικόνα του.

Όπως σε κάθε προηγούμενη σελίδα για τις βασικές λειτουργίες, κάθε ταξιδιωτική λίστα έχει τις επιλογές “Edit” και “Delete”. Το κουμπί της επεξεργασίας ανακατευθύνει σε μια σελίδα φόρμας με όνομα “Edit Travel List” όπου μπορούν να επεξεργαστούν την ταξιδιωτική λίστα με όλα τα προηγούμενα στοιχεία να είναι συμπληρωμένα και τα ρούχα επιλεγμένα. Με το κουμπί της διαγραφής, η ταξιδιωτική λίστα αφαιρείται απευθείας χωρίς όμως να επηρεάζονται τα ρούχα που περιλαμβάνει. Σε παρένθεση δεξιά του τίτλου “Travel Lists” παρατηρείται ο συνολικός αριθμός δημιουργημένων ταξιδιωτικών λιστών.

Η επιλογή Create Travel Lists οδηγεί στη φόρμα “Create a Travel List” όπου μπορούν να δημιουργήσουν μια νέα ταξιδιωτική λίστα συμπληρώνοντας πεδία όπως ονομασία, προορισμός, ημερομηνίες αναχώρησης και επιστροφής, περιγραφή και επιλέγοντας τα επιθυμητά ρούχα όπως και στα σύνολα. Για την εύρεση ενός ρούχου έχει συμπεριληφθεί η μηχανή αναζήτησης με βάση την ονομασία, κατηγορία, χρώμα και εποχή χρήσης. Κάθε ρούχο στο πεδίο επιλογής περιλαμβάνεται σε ένα γκρι πλαίσιο το οποίο αλλάζει σε μπεζ όταν επιλεγεί.



**Εικόνα 28:** Φόρμα δημιουργίας ταξιδιωτικής λίστας με τα πεδία συμπλήρωσης, τα φίλτρα αναζήτησης ρούχων και δυο μαρκαρισμένα επιλεγμένα αντικείμενα.

### 3.4.5 Στυλιστικές Επιλογές

Οι στυλιστικές επιλογές για την εφαρμογή “Digital Wardrobe” συνδυάζουν μοντέρνα εργαλεία και αισθητική για να προσφέρουν ένα ευχάριστο και λειτουργικό περιβάλλον χρήσης [62]. Τα εργαλεία που επιλέχθηκαν διακρίνονται σε διάφορα σημεία της διαδικτυακής εφαρμογής, τόσο στην ταπετσαρία και τα λογότυπα όσο και στη γραμματοσειρά και τη χρωματική παλέτα που χρησιμοποιούνται. Στις εικόνες του κεφαλαίου μπορούν να διακριθούν όλες οι στυλιστικές επιλογές που θα αναλυθούν.

Για τα εικονίδια στα κουμπιά “Edit”, “Create ..”, “Delete”, “Update” και “Search Filters”, στις λειτουργίες “Home”, “Sign Out”, “Sign In” και “Sign Up”, και στα μέσα κοινωνικής δικτύωσης της εφαρμογής, χρησιμοποιήθηκε το εργαλείο **Font Awesome**. Συγκεκριμένα, επιλέχθηκαν εικονίδια, κινούμενα και μη, από την **δωρεάν** έκδοση του εργαλείου με απλή αναζήτηση στην ιστοσελίδα του. Για την συμπερίληψή τους στο εργαλείο, αντιγράφηκε ο

παρεχόμενος κώδικας HTML στα κατάλληλα αρχεία του components και η εντολή “`@import "~@fortawesome/fontawesome-free/css/all.min.css";`” στο αρχείο App.css. Για την μορφή των κουμπιών, χρησιμοποιήθηκε το εργαλείο **CSS Button Generator** της istoselidias Front-end Tools, όπου από εκεί επιλέχθηκε το κατάλληλο με τον CSS κώδικά του για όλες τις λειτουργίες.

Σε ορισμένα σημεία της εφαρμογής παρατηρείται μια διαφορετική γραμματοσειρά από την πιο κοινή σε χρήση Arial. Αυτή η γραμματοσειρά διακρίνεται σε όλα τα κουμπιά, στην επικεφαλίδα, στο υποσέλιδο και σε όλες τις βασικές λειτουργίες. Επιλέχθηκε αυτή με ονομασία Handwriting από την κατηγορία **Fancy** του εργαλείου **Font Generator** του **Quicktools By Piesart**. Ο λόγος για τον οποίο επιλέχθηκε η συγκεκριμένη είναι γιατί προσθέτει μια καλλιτεχνική πινελιά στη συνολική εικόνα της εφαρμογής μας. Για παράδειγμα, η μορφή της γραμματοσειράς είναι η εξής “**DIGITAL WARDROBE**”. Για την χρωματική παλέτα που εμφανίζεται σε όλες τις σελίδες, επιλέχθηκαν τα χρώματα από την σελίδα **ImageColorPicker** με την χρήση της ταπετσαρίας. Τα κύρια χρώματα που εμφανίζονται είναι το μπεζ με κωδικό **#eed9c4** και το καφέ με κωδικό **#633200**, καθώς και το άσπρο και μαύρο. Τα χρώματα μπεζ και καφέ επιλέχθηκαν χάρη στην ασφάλεια και άνεση που προσφέρουν στα μάτια των χρηστών αλλά και γιατί είναι από τα πιο συχνά χρώματα ντουλαπών.

Για το λογότυπο της εφαρμογής (αριστερά στην επικεφαλίδα και favicon), την ταπετσαρία και το background σε επικεφαλίδα και υποσέλιδο, χρησιμοποιήθηκε το εργαλείο παραγωγής AI εικόνων χωρίς πνευματικά δικαιώματα, **DALL-E** του **ChatGPT**. Συγκεκριμένα, για το λογότυπο ζητήθηκε να έχει συνδυασμό λευκών, χρυσών και καφέ χρωμάτων, και να απεικονίζει μια μοντέρνα ντουλάπα με το ένα της φύλλο ανοιχτό με ένα κρεμασμένο ρούχο. Για την ταπετσαρία ζητήθηκε ένα background που θα αναγράφει τον τίτλο “Digital Wardrobe” και θα περιλαμβάνει μία μίξη ανδρικών και γυναικείων ειδών ένδυσης. Για την επικεφαλίδα και το υποσέλιδο ζητήθηκε να παράξει ένα background το οποίο θα μοιάζει με την μορφή ενός μασίφ μπεζ ξύλου, υλικό που χρησιμοποιείται στις περισσότερες ντουλάπες σήμερα.

Η ανάπτυξη του υποσέλιδου βασίστηκε σε ένα YouTube βίντεο του χρήστη The WebShala. Η δομή, η ανταπόκρισή του στην κίνηση και η στοίχιση των στηλών και όλων των αντικειμένων που περιλαμβάνει κάθε μία από αυτές, μας οδήγησε την επιλογή του. Χρησιμοποιήθηκε ο παρεχόμενος κώδικας τον οποίο εμείς προσαρμόσαμε στα ζητούμενα και δεδομένα της εφαρμογής μας.

### 3.4.6 Δοκιμή και Έλεγχος

Για τη δοκιμή, δημιουργήθηκε ένα draft προφίλ χρήστη με όλα τα πεδία εγγραφής συμπληρωμένα, ώστε να έχουμε μια ολοκληρωμένη εικόνα της εμπειρίας χρήστη σε κάθε λειτουργία. Μέσω αυτού, πραγματοποιήθηκαν δοκιμές σε όλες τις βασικές λειτουργίες της εφαρμογής, δηλαδή την προσθήκη, επεξεργασία και διαγραφή ρούχων, συνόλων, περιστάσεων, ταξιδιωτικών λιστών και προφίλ χρηστών. Η επικοινωνία με το Back-End και η σωστή επεξεργασία των δεδομένων στη βάση MongoDB ελέγχονταν σε κάθε βήμα, διασφαλίζοντας την ομαλή ανταλλαγή δεδομένων, με ταυτόχρονη προσαρμογή του κώδικα στο Back-End όπου χρειαζόταν [28].

Για τον εντοπισμό και την επίλυση σφαλμάτων, χρησιμοποιήθηκε η επιλογή **Inspect** του Google Chrome, και πιο συγκεκριμένα οι καρτέλες **Console** και **Network**. Μέσω της Console, εντοπίζονταν σφάλματα στον κώδικα JavaScript και τυχόν ασυνέπειες κατά την αλληλεπίδραση των Back-End και Front-End, ενώ η καρτέλα Network παρείχε λεπτομέρειες σχετικά με τα αιτήματα HTTP και την ανταπόκριση της βάσης δεδομένων, βοηθώντας στην ανάλυση της απόδοσης και διόρθωσης σφαλμάτων [22].

Επιπλέον, στο στάδιο του σχεδιασμού της διεπαφής χρήστη, πραγματοποιήθηκε εκτενής πειραματισμός μέσω του αρχείου **App.css**. Δοκιμάστηκαν διάφορα χρώματα, εικόνες, και η δομή του περιεχομένου, ώστε να καταλήξουμε στην τελική εμφάνιση της εφαρμογής. Με τη συνεχή βελτίωση του σχεδιασμού και την προσαρμογή των χρωμάτων και περιεχομένων, επιτεύχθηκε το επιθυμητό οπτικό αποτέλεσμα και η βέλτιστη εμπειρία χρήστη [72].

## 3.5 Προκλήσεις και Τεχνικά Εμπόδια

Κατά τη διάρκεια της ανάπτυξης του Front-End, προέκυψαν και αντιμετωπίστηκαν διάφορες προκλήσεις και τεχνικά εμπόδια, τα οποία ήταν καθοριστικά για την αποτελεσματική λειτουργία και αισθητική του τελικού προϊόντος. Οι βασικές προκλήσεις αφορούσαν την προσαρμογή της διεπαφής χρήστη, την διασφάλιση της ασφάλειας δεδομένων, καθώς και την ενσωμάτωση εξωτερικών βιβλιοθηκών και εργαλείων. Τα τεχνικά εμπόδια αποτελούσαν η διαχείριση ασύγχρονων αιτημάτων, κατάστασης και σφαλμάτων. Παρακάτω αυτά θα αναλυθούν περαιτέρω, με αναφορά στον τρόπο αντιμετώπισή τους, διασφαλίζοντας την ομαλή και ασφαλή λειτουργία του Front-End της εφαρμογής.

### 3.5.1 Προκλήσεις και Λύσεις

Η προσαρμογή της διεπαφής χρήστη ώστε να είναι προσιτή και εύκολη στη χρήση προς όλους τους χρήστες, ανεξαρτήτως ηλικίας και εμπειρίας, αποτέλεσε βασική πρόκληση. Για αυτό τον λόγο επιλέχθηκε κατάλληλος συνδυασμός χρωμάτων, εικονιδίων, γραμματοσειρών και διατάξεων, ώστε η εφαρμογή να έχει μια καθαρή και απλή δομή. Επιπλέον, έγινε πειραματισμός με την CSS ώστε από καθοδήγηση διαφόρων οδηγών σχεδιασμού απλών και εύχρηστων διαδικτυακών εφαρμογών στο διαδίκτυο.

Η εξασφάλιση ότι τα δεδομένα των χρηστών παραμένουν ασφαλή κατά την επικοινωνία μεταξύ Front-End και Back-End απαιτούσε μια ασφαλή αρχιτεκτονική και σωστή διαχείριση των αιτημάτων. Αυτό επιτεύχθηκε με την χρήση ασφαλών αιτημάτων, την εφαρμογή αυθεντικοποίησης και εξουσιοδότησης με JWT tokens, καθώς και την τακτική επικύρωση δεδομένων μέσω του express-validator. Επιπλέον, περιορίστηκε η πρόσβαση σε ορισμένες λειτουργίες μόνο για συνδεδεμένους χρήστες.

Η επιλογή των κατάλληλων εργαλείων και βιβλιοθηκών που να ταιριάζουν στο στυλ και τις ανάγκες της εφαρμογής αποτέλεσε πρόκληση. Για παράδειγμα, η χρήση του Font Awesome για εικονίδια απαιτούσε καλή προσαρμογή στον σχεδιασμό. Η προσαρμογή αυτού αλλά και υπόλοιπων εργαλείων και βιβλιοθηκών έγινε προσεκτικά μέσω των CSS και JavaScript, και πραγματοποιούνταν συνεχής έλεγχος της συνολικής εμφάνισης της ιστοσελίδας.

### 3.5.2 Τεχνικά Εμπόδια και Λύσεις

Η διαχείριση των ασύγχρονων αιτημάτων απαιτεί καλό συντονισμό για να διασφαλιστεί η ταχύτητα και ακρίβεια της επικοινωνίας με το Back-End. Προβλήματα όπως τα λανθασμένα δεδομένα ή καθυστερημένες απαντήσεις από τον διακομιστή μπορούν να προκαλέσουν δυσλειτουργίες αλλά και δυσαρέσκεια στους χρήστες. Για την επίλυση αυτών των προβλημάτων χρησιμοποιήθηκε το **async/await** ώστε να ελεγχθεί καλύτερα η ροή των αιτημάτων. Επιπλέον, με τη χρήση του **Axios** διευκολύνθηκε η διαχείριση των HTTP αιτημάτων και η διαχείριση των απαντήσεων (responses).

Η διατήρηση της κατάστασης (state) των δεδομένων στο React.js αποτελεί βασικό τεχνικό εμπόδιο, ειδικά όταν η εφαρμογή βασίζεται σε πολλές και ταυτόχρονες αλλαγές δεδομένων. Γι' αυτό τον λόγο χρησιμοποιήθηκαν hooks, όπως τα **useState** και **useEffect**, για την ενημέρωση και διατήρηση των δεδομένων της εφαρμογής. Επίσης, έγινε σωστή οργάνωση της λογικής εντός όλων των components ώστε να μειωθεί η πιθανότητα ασυνεπειών στα δεδομένα.

Η παρακολούθηση σφαλμάτων και η άμεση διόρθωσή τους κατά τη διάρκεια της ανάπτυξης αποτελούν εμπόδιο για την ομαλή λειτουργία της εφαρμογής, ειδικά στην επικοινωνία με το Back-End. Στην ανάπτυξη της εφαρμογής μας, ο έλεγχος έγινε μέσω των **DevTools** του Google Chrome, όπου οι καρτέλες **Console** και **Network** βοήθησαν στον εντοπισμό των σφαλμάτων. Οι διορθώσεις έγιναν με τη χρήση κατάλληλων μηνυμάτων σφάλματος στην πλευρά του Back-End και τις βελτιώσεις στην επεξεργασία των απαντήσεων από το Front-End.

### 3.6 Συμπεράσματα

Η ανάπτυξη του Front-End του Digital Wardrobe επικεντρώθηκε στην ευχρηστία, την προσβασιμότητα και την αισθητική της διεπαφής χρήστη, με σκοπό τη δημιουργία μιας ομαλής και διαδραστικής εμπειρίας χρήστη. Η επιλογή εργαλείων και τεχνολογιών, όπως το React.js, το Axios αλλά και αυτών που αποτέλεσαν τις στιλιστικές επιλογές, επέτρεψε την ανάπτυξη μιας δυναμικής εφαρμογής, που προσφέρει ταχύτητα και αποτελεσματικότητα στη διαχείριση προφίλ, ρούχων, συνόλων, περιστάσεων και ταξιδιωτικών λιστών, αλλά και ασφάλεια και προστασία στα δεδομένα χρηστών.

Κατά την υλοποίηση του Front-End, αντιμετωπίστηκαν προκλήσεις τόσο στον σχεδιασμό όσο και στη διασφάλιση της ορθής επικοινωνίας με το Back-End, αλλά και στη διατήρηση μιας συνεπούς αισθητικής. Η χρήση του React.js για την κατασκευή επαναχρησιμοποιούμενων στοιχείων (components), και η σύνδεση με το Back-End μέσω Axios συνέβαλαν στην αξιόπιστη και γρήγορη απόκριση της εφαρμογής, παρέχοντας στους χρήστες μια διαισθητική και ευχάριστη εμπειρία, ενώ οι στυλιστικές επιλογές όπως η ταπετσαρία, τα Font Awesome εικονίδια και η χρωματική παλέτα συνέβαλαν στη σχεδίαση μιας ευχάριστης και απλής διεπαφής χρήστη.

Με την ολοκλήρωση του Front-End, η εφαρμογή ενισχύθηκε ως προς τη λειτουργικότητα και την προσβασιμότητα, δημιουργώντας ένα περιβάλλον που ανταποκρίνεται άμεσα στις ανάγκες των χρηστών. Αποτέλεσε το τελικό βήμα υλοποίησης του τελικού προϊόντος με όνομα “Digital Wardrobe”. Με τη χρήση κατάλληλων βιβλιοθηκών και εργαλείων εξασφαλίστηκε η αισθητική συνοχή, η ομαλή λειτουργία και η εύκολη πλοήγηση, καθιστώντας την εφαρμογή εύχρηστη και προσβάσιμη για όλους τους χρήστες ανεξαρτήτως ηλικίας και εμπειρίας.



## 4 ΣΥΖΗΤΗΣΗ, ΒΕΛΤΙΩΣΕΙΣ ΚΑΙ ΣΥΜΠΕΡΑΣΜΑΤΑ

Σε αυτό το κεφάλαιο θα παρουσιαστούν οι **τελικές σκέψεις** σχετικά με την ανάπτυξη και τη λειτουργικότητα του “Digital Wardrobe”. Αρχικά, θα συζητηθούν τα βασικά χαρακτηριστικά και οι επιλογές σχεδιασμού της εφαρμογής, καθώς και οι προκλήσεις που αντιμετωπίστηκαν κατά την ανάπτυξη, και η συμβολή στη βιώσιμη διαχείριση της γκαρνταρόμπας. Στη συνέχεια, θα παρουσιαστούν οι πιθανές βελτιώσεις που θα μπορούσαν να ενισχύσουν την εμπειρία χρήστη και να επεκτείνουν τις δυνατότητες της εφαρμογής ώστε να καλύπτει την κάθε πιθανή ανάγκη των χρηστών. Τέλος, θα παρουσιαστούν τα τελικά συμπεράσματα που συνοψίζουν την όλη εμπειρία ανάπτυξης, από την αρχική ιδέα έως την ολοκλήρωση, υπογραμμίζοντας τα επιτεύγματα, τις δυσκολίες και τα μαθήματα που αποκομίστηκαν από το συνολικό έργο.

### 4.1 Συζήτηση

Η όλη διαδικασία ανάπτυξης της διαδικτυακής εφαρμογής **“Digital Wardrobe”** μας βοήθησε να ανακαλύψουμε τις βασικές πτυχές που είναι αναγκαίες για τη δημιουργία μιας αποτελεσματικής εφαρμογής διαχείρισης και οργάνωσης μιας ντουλάπας. Μέσα από την ακολουθία βημάτων ανάπτυξης της βάσης δεδομένων MongoDB, του Back-End και του Front-End, το τελικό έργο ανταποκρίθηκε στον αρχικό στόχο του να προσφέρει στους χρήστες μια πρακτική και εύχρηστη πλατφόρμα για την καλύτερη οργάνωση της ντουλάπας τους αλλά και της σωστής διαχείρισης των ειδών ένδυσης σε αυτήν. Η εφαρμογή με τις βασικές λειτουργίες που παρέχει, δίνει τις λύσεις σε προβλήματα της καθημερινότητας των ανθρώπων που δυσκολεύονται με τη διαχείριση του χρόνου και την ακαταστασία σε μια ντουλάπα, και έχουν την ανάγκη για βοήθεια στην άθηση ή ανύψωση του οργανωτικού πνεύματος.

Η ανάπτυξη του Back-End εξασφάλισε την αποτελεσματική διαχείριση δεδομένων και την ασφαλή και αδιάκοπη επικοινωνία μεταξύ των χρηστών και της βάσης δεδομένων MongoDB. Η χρήση σύγχρονων τεχνολογιών και βιβλιοθηκών όπως τα Node.js, Express.js και Mongoose επέτρεψαν την υλοποίηση ενός αποδοτικού συστήματος, ενώ η ασφάλεια των δεδομένων ενισχύθηκε με εργαλεία όπως το JWT. Παράλληλα, στο Front-End, το React.js προσέφερε τη δυνατότητα δημιουργίας μιας διαδραστικής διεπαφής χρήστη, επιτρέποντας την ομαλή αλληλεπίδραση με τις λειτουργίες της εφαρμογής. Το Axios συνέβαλε σημαντικά στη συνεχή επικοινωνία με το Back-End, ενισχύοντας την ταχύτητα και την αποδοτικότητα της εφαρμογής.

Στη διάρκεια της ανάπτυξης του “Digital Wardrobe”, αντιμετωπίστηκαν ποικίλες προκλήσεις και τεχνικά εμπόδια που επηρέασαν τα βήματα που ακολουθήθηκαν για την εξασφάλιση της λειτουργικότητας και της αποδοτικότητας του συστήματος. Από την πλευρά του Back-End, προκλήσεις όπως η ασύγχρονη επικοινωνία με τη βάση δεδομένων MongoDB και η ασφάλεια των δεδομένων των χρηστών απαιτούσαν ειδική διαχείριση και χρήση ασφαλών βιβλιοθηκών. Επιπλέον, το Front-End χρειάστηκε προσαρμογές για να διασφαλιστεί η ομαλή και απρόσκοπη επικοινωνία με το Back-End, ειδικά κατά τη διαχείριση πολλαπλών αιτημάτων. Η σχεδίαση της διεπαφής βασίστηκε στην υλοποίηση ενός φιλικού και ταυτόχρονα διαδραστικού περιβάλλοντος για τους χρήστες, κάτι που απαιτούσε συνεχή δοκιμή και έλεγχο του σχεδιασμού και των λειτουργιών για την επίτευξη του αρχικού στόχου. Παρά τις δυσκολίες, οι προκλήσεις και τα εμπόδια αυτά αλλά και η διαδικασία εύρεσης των καλύτερων δυνατών λύσεων οδήγησαν σε βελτιώσεις που ενίσχυσαν τόσο την αποδοτικότητα όσο και την αξιοπιστία της εφαρμογής.

Οι προτεινόμενες πρόσθετες λειτουργίες και σχεδιαστικές βελτιώσεις, που αναλύονται στην επόμενη ενότητα, δείχνουν πώς η διαδικτυακή εφαρμογή μας θα μπορούσε να επεκταθεί στο μέλλον και να εξυπηρετεί πολλές περισσότερες ανάγκες των χρηστών. Η σύνδεση με την τεχνητή νοημοσύνη (AI) για την χρήση της σε πολλές λειτουργίες, η δημιουργία μιας κοινότητας τύπου μέσων κοινωνικής δικτύωσης και η ενσωμάτωση συνεργαζόμενων επιχειρήσεων ή υπηρεσιών, θα μπορούσαν να εμπλουτίσουν τις δυνατότητες της εφαρμογής και να την καταστήσουν ακόμα πιο ολοκληρωμένη και χρήσιμη για το υπάρχον κοινό της, δημιουργώντας παράλληλα το μονοπάτι για την “εισροή” νέων χρηστών. Έτσι, το “Digital Wardrobe” έχει τις βάσεις για να εξελιχθεί σε ένα πλήρες ψηφιακό εργαλείο διαχείρισης της γκαρνταρόμπας, παρέχοντας μια πιο ολοκληρωμένη και απλά δομημένη εμπειρία στους χρήστες της.

Συνοπτικά, η ανάπτυξη της διαδικτυακής εφαρμογής **“Digital Wardrobe”** έθεσε τις βάσεις για την περαιτέρω βελτίωση της με νέες δυνατότητες, και λειτουργίες διατηρώντας ως στόχο τη βελτίωση της εμπειρίας του χρήστη και τη διευκόλυνση της καθημερινότητας με τον καλύτερο δυνατό τρόπο. Η εφαρμογή, παρότι προσφέρει κάποιες βασικές λειτουργίες για την οργάνωση ειδών ένδυσης, έχει δυνατότητες επέκτασης και αναβάθμισης που θα την καταστήσουν ακόμα πιο ευέλικτη και χρήσιμη για τους χρήστες στο μέλλον.

## 4.2 Βελτιώσεις

Η εφαρμογή **Digital Wardrobe** έχει ήδη καταφέρει να καλύψει σημαντικές ανάγκες των χρηστών με τις λειτουργίες που ήδη προσφέρονται, όπως η οργάνωση των ειδών ένδυσης της

γκαρνταρόμπας, η δημιουργία συνόλων, η δημιουργία και διαχείριση ταξιδιωτικών λιστών και η εύκολη κατηγοριοποίηση συνόλων βάση περιστάσεων. Αυτές οι δυνατότητες διευκολύνουν τους χρήστες στην καθημερινή τους ζωή και τους προσφέρουν ένα εργαλείο που ενισχύει το πρακτικό και οργανωτικό πνεύμα για την ορθή διαχείριση των προσωπικών τους ειδών. Ωστόσο, με την ενσωμάτωση ορισμένων πρόσθετων λειτουργιών και κάποιων σχεδιαστικών **βελτιώσεων**, η διαδικτυακή εφαρμογή μπορεί να ενισχυθεί ακόμη περισσότερο, προσφέροντας μια ακόμη πιο ολοκληρωμένη, ευχάριστη και εμπλουτισμένη εμπειρία χρήστη. Αυτές οι βελτιώσεις θα πρέπει να εφαρμοστούν πάντα με την λογική της απλότητας και της εμπειρίας χρηστών. Παρακάτω θα αναλυθούν αυτές οι βελτιώσεις θεωρητικά, χωρίς να γίνει περαιτέρω εμβάθυνση στους τρόπους υλοποίησης.

**Κοινότητα Χρηστών (Community):** Δημιουργία μιας κοινότητας-μπλογκ εντός της διαδικτυακής εφαρμογής και συγκεκριμένα προσβάσιμη από την επικεφαλίδα, όπου οι χρήστες μπορούν να μοιράζονται σύνολα και ρούχα μεταξύ τους, εμπνέοντας ο ένας τον άλλο και επιτρέποντας την ανακάλυψη νέων ιδεών για το προσωπικό τους στυλ. Θα μπορούσε να λειτουργεί σαν κάποιο μέσο κοινωνικής δικτύωσης όπου μπορούν οι χρήστες να ακολουθήσουν ο ένας τον άλλο και να σχολιάζουν τα σύνολα και ρούχα τους.

**Αυτόματη κατηγοριοποίηση ρούχων με image recognition:** Προσθήκη λειτουργίας που χρησιμοποιεί τεχνητή νοημοσύνη για να αναγνωρίζει τα ρούχα κατά το ανέβασμα εικόνων, καταχωρώντας αυτόματα κατηγορία, χρώμα και άλλες σχετικές λεπτομέρειες. Βέβαια, δεν θα γίνεται αυτόματο ανέβασμα στο προφίλ ώστε οι χρήστες να μπορούν να επεξεργαστούν τα πεδία πριν την υποβολή.

**Πρόταση συνόλων μέσω AI:** Ανάπτυξη λειτουργίας που προτείνει σύνολα με βάση τα ρούχα που έχουν ήδη ανεβάσει οι χρήστες, τις χρωματικές τους προτιμήσεις ή το ιστορικό των επιλεγμένων συνδυασμών, τον καιρό της ημέρας και την περίσταση που έχει επιλεγεί, χρησιμοποιώντας τεχνητή νοημοσύνη. Αυτή η λειτουργία μπορεί να βοηθήσει ακόμη περισσότερο τους χρήστες στο να οδηγηθούν στο στυλ που τους ταιριάζει καλύτερα.

**Προτάσεις ρούχων για ταξίδια:** Δημιουργία δυνατότητας που προτείνει κατάλληλα ρούχα για συγκεκριμένους προορισμούς και καιρικές συνθήκες, βοηθώντας τους χρήστες να προετοιμάζουν την ταξιδιωτική τους λίστα ανάλογα με τον προορισμό και την περίοδο του ταξιδιού. Για παράδειγμα, εάν το ταξίδι έχει προορισμό κάποιο νησί της Ελλάδας και η περίοδος ταξιδιού είναι δύο εβδομάδες του Αυγούστου, τότε να του προτείνει τα κατάλληλα καλοκαιρινά ρούχα με βάση τα δεδομένα καιρού για το συγκεκριμένο νησί, ενώ παράλληλα να του δίνει το σωστό αριθμό ρούχων που θα πρέπει να πάρει μαζί του ώστε να κάνει σωστή διαχείριση χώρου βαλίτσας.

**Σύνδεση με AI για προώθηση προτεινόμενων προϊόντων:** Ενσωμάτωση AI που προτείνει προϊόντα από συνεργαζόμενα brands ή καταστήματα, σε περίπτωση που γίνει κάποια συνεργασία ή συμφωνία με τρίτο, με βάση τα ρούχα που διαθέτουν οι χρήστες στην ντουλάπα τους, τα σύνολα με τα οποία μπορεί να ταιριάζει ένα νέο ρούχο, επιτρέποντας προσωποποιημένες προτάσεις αγορών.

**Δυνατότητα αλλαγής γλώσσας:** Μία λειτουργική προσθήκη θα ήταν η δυνατότητα επιλογής πολλαπλών γλωσσών από το υποσέλιδο, ώστε η εφαρμογή να είναι προσιτή σε ένα ευρύτερο κοινό, ανεξαρτήτως γλωσσικών προτιμήσεων. Καθώς η γλώσσα που χρησιμοποιείται είναι τα αγγλικά, θα μπορούσαν να συμπεριληφθούν και άλλες ώστε να μπορούν να χρησιμοποιούν την εφαρμογή και άλλοι άνθρωποι που μπορεί να προτιμούν να την λειτουργούν στη γλώσσα τους.

**Λειτουργία λίστας επιθυμιών:** Δυνατότητα δημιουργίας λίστας επιθυμιών (wishlist) όπου οι χρήστες μπορούν να αποθηκεύουν τα ρούχα που θέλουν να αγοράσουν, προσθέτοντας εικόνες και λεπτομέρειες όπως αν ανεβάζουν ένα ρούχο στο προφίλ τους, αλλά και έναν σύνδεσμο από το ηλεκτρονικό μαγαζί στο οποίο διατίθεται. Επιπλέον, η λίστα επιθυμιών θα προσφέρει την επιλογή μεταφοράς των αντικειμένων στα κανονικά ρούχα της γκαρνταρόμπας όταν αυτά αγοραστούν, χωρίς να απαιτείται νέα καταχώρηση. Σημαντικό στοιχείο θα είναι μέσω της τεχνητής νοημοσύνης να αναγνωρίζεται από την εφαρμογή εάν κάποιο είδος ένδυσης στη λίστα επιθυμιών είναι περιττό, δηλαδή να το συγκρίνει με τα ανεβασμένα ρούχα στο προφίλ των χρηστών και να κρίνει εάν υπάρχουν άλλα παρόμοια με αυτό.

**Προγραμματισμός εβδομαδιαίων συνόλων:** Δυνατότητα δημιουργίας εβδομαδιαίου προγράμματος για τους χρήστες, με σύνολα για κάθε ημέρα της εβδομάδας, επιτρέποντάς τους να οργανώνουν το ντύσιμό τους. Για παράδειγμα, μια ομάδα ανθρώπων για τους οποίους θα ήταν χρήσιμη αυτή η λειτουργία είναι για εκείνους που μπορεί να ασχολούνται με κάποιο επάγγελμα αυστηρού ντυσίματος όπως αυτό της δουλειάς γραφείου σε μια ασφαλιστική εταιρεία.

**Εντοπισμός συχνής χρήσης ρούχων ή συνόλων:** Προσθήκη λειτουργίας που επιτρέπει στους χρήστες να επιλέγουν ποια ρούχα ή σύνολα φόρεσαν κάθε μέρα και να λαμβάνουν ειδοποιήσεις εάν κάποιο ρούχο έχει χρησιμοποιηθεί πολλές φορές (τουλάχιστον τρεις) την τελευταία εβδομάδα, προσφέροντας μια πιο ισορροπημένη χρήση της γκαρνταρόμπας ώστε όλα τα είδη ένδυσης να φοριούνται με την ίδια συχνότητα. Σε σύνδεση με την πρόταση συνόλων μέσω τεχνητής νοημοσύνης, θα μπορούσε να η εφαρμογή να προτείνει σύνολα με ρούχα που σπανίως φοριούνται από τους χρήστες.

**Διαχείριση ρούχων μέσω barcode/QR code:** Δυνατότητα προσθήκης ρούχων με την σάρωση των barcode ή QR codes που συνήθως περιλαμβάνονται στα καρτελάκια τους, η

οποία θα αναγνωρίζει το προϊόν και θα προσθέτει αυτόματα τις πληροφορίες του (π.χ., μάρκα, χρώμα, κατηγορία). Μπορεί αντίστοιχα να ενσωματωθεί και στη λειτουργία wishlist, όπου οι χρήστες σαρώνουν το κωδικό του ρούχου στο φυσικό κατάστημα και αυτό αυτομάτως μετακινείται στη λίστα επιθυμιών τους.

**Λειτουργία οργάνωσης και διαχείρισης ντουλάπας με χρονική κατηγοριοποίηση:** Δημιουργία επιλογών ταξινόμησης με βάση τη συχνότητα χρήσης, π.χ. τα ρούχα που δεν έχουν φορεθεί για μεγάλο χρονικό διάστημα (παραπάνω από δύο εβδομάδες) να μεταφέρονται σε μια κατηγορία, υπό της καρτέλας “Clothes”, με όνομα “Rare Use” (σπανίως σε χρήση). Αυτά που χρησιμοποιούνται πιο συχνά από το σύνηθες μπορούν να μεταφέρονται σε μια άλλη κατηγορία με όνομα “Everyday Use” (καθημερινή χρήση).

**Λειτουργία στατιστικών ντουλάπας:** Δημιουργία μιας λειτουργίας κατά την οποία η εφαρμογή συγκρατεί στη μνήμη τη συχνότητα χρήσης ειδών ένδυσης και συνόλων, τα ρούχα που προστέθηκαν ή αφαιρέθηκαν τον τελευταίο μήνα και τα ρούχα που προστέθηκαν και αγοράστηκαν από τη λίστα επιθυμιών.

**Αυτόματη παρακολούθηση φθοράς και κατάστασης ρούχων:** Χρήση AI για την πρόταση ανανέωσης ή αντικατάστασης ρούχων όταν παρατηρηθεί ότι συγκεκριμένα είδη ένδυσης χρησιμοποιούνται πολύ συχνά, έχουν φθορές ή τα φοράνε πολλά χρόνια (με σύγκριση φωτογραφιών του ρούχου πριν και μετά).

**Αυτόματη κατηγοριοποίηση ρούχων με βάση τα χρώματα:** Ενσωμάτωση λειτουργίας αναγνώρισης χρωμάτων ώστε τα ρούχα να ομαδοποιούνται με βάση το κύριο χρώμα τους (χωρίς να λαμβάνονται υπόψη τυχόν στάμπες), δίνοντας έτσι τη δυνατότητα στους χρήστες να φίλτράρουν τα ρούχα τους ανά χρωματικές παλέτες.

**Λειτουργία ψηφιακής προσομοίωσης συνόλων σε AR:** Ενσωμάτωση τεχνολογίας AR (Augmented Reality) για τη δημιουργία μιας εικονικής δοκιμής συνόλων. Οι χρήστες θα μπορούν να “δοκιμάσουν” τα πρόχειρα σύνολα τους, πριν την αποθήκευση, σε μια επαυξημένη προσομοίωση και να έχουν μια καλύτερη εικόνα για το πώς θα δείχνουν συνδυασμένα μαζί. Μπορεί επίσης να χρησιμοποιηθεί στα ρούχα της λίστας επιθυμιών ώστε να μπορούν οι χρήστες να κρίνουν εάν τους ταιριάζουν (σε περίπτωση που δεν υπάρχει η δυνατότητα φυσικής δοκιμής τους).

**Ειδοποίησεις για πλύσιμο και συντήρηση ρούχων:** Δυνατότητα προσθήκης ειδοποιήσεων για πλύσιμο ή συντήρηση συγκεκριμένων ρούχων, ανάλογα με τον αριθμό των φορών που τα έχει φορέσει. Για παράδειγμα, εάν κάποια μπλούζα έχει φορεθεί παραπάνω από πέντε φορές θα πρέπει να πλυνθεί τόσο για να συντηρηθεί το ρούχο όσο και για να είναι καθαρό.

**Διαδραστικός χάρτης ντουλάπας:** Μια 3D οπτικοποίηση της ντουλάπας σε μορφή διαγράμματος ή “χάρτη”, όπου τα ρούχα και τα αξεσουάρ θα παρουσιάζονται με εικονίδια που θα δείχνουν τη θέση τους στην ντουλάπα και την κατηγορία τους, διευκολύνοντας έτσι τη γρήγορη εύρεση και επιλογή τους.

**Επικοινωνία με υπηρεσίες στεγνού καθαρισμού ή ράφτες:** Ενσωμάτωση πληροφοριών για υπηρεσίες στεγνού καθαρίσματος ή ράφτες με βάση την τοποθεσία των χρηστών. Για κάθε υπηρεσία θα εμφανίζεται η οδός του καταστήματος αλλά και οι τρόπου επικοινωνίας.

**Επικοινωνία με οιμάδες δωρεάς ρούχων ή τοποθεσία κουτιών ανακύκλωσης:** Παροχή μιας λίστας επικοινωνίας με οιμάδες που συλλέγουν ρούχα για δωρεά σε ανθρώπους που τα έχουν ανάγκη. Παράλληλα, θα μπορούσε να υπάρχει μια λίστα με τα κοντινά κουτιά ανακύκλωσης ρούχων βάση τοποθεσίας των χρηστών. Αυτή η λειτουργία θα είχε σημασία σε περίπτωση που κάποιοι χρήστες θέλουν να κάνουν κάποια αναδιάταξη ή ξεσκαρτάρισμα στην γκαρνταρόμπα τους.

**Υπενθυμίσεις για εκπτώσεις σε προϊόντα wishlist:** Εάν οι χρήστες έχουν δημιουργήσει μια λίστα επιθυμιών, μπορούν να ειδοποιούνται για εκπτώσεις ή προσφορές σε αυτά τα ρούχα από συνεργαζόμενα καταστήματα ή brands, παρέχοντας έτσι εξοικονόμηση χρόνου και χρήματος, ενημερώνοντας τους σε περίπτωση που επιθυμούν να προβούν σε αγορά. Θα μπορούσε, μέσω της συνεργασίας, να δημιουργηθεί κάποιος εκπτωτικός κωδικός εάν η αγορά πραγματοποιείται από χρήστες της εφαρμογής.

### 4.3 Τελικά Συμπεράσματα

Η εφαρμογή “**Digital Wardrobe**” σχεδιάστηκε με στόχο να βοηθήσει τους χρήστες στη σωστή διαχείριση της γκαρνταρόμπας τους, προσφέροντας ένα πλήρως δομημένο ψηφιακό εργαλείο για την οργάνωση των ρούχων, τη δημιουργία συνόλων, την κατηγοριοποίηση συνόλων ανά περιστάσεις και τη διευκόλυνση της προετοιμασίας ταξιδιών με τις λίστες. Μέσα από τη δυνατότητα προβολής, κατηγοριοποίησης και εύκολης διαχείρισης των ρούχων, η εφαρμογή μας επιδιώκει να εξαλείψει το πρόβλημα της ακαταστασίας στην ντουλάπα και της δυσκολίας στην επιλογή ενός συνόλου. Παράλληλα, δυνατότητες όπως η δημιουργία και αποθήκευση συνόλων και η οργάνωση ταξιδιωτικών λιστών παρέχουν πολλά οφέλη στους χρήστες, βελτιώνοντας την καθημερινότητα και την σχέση τους με την γκαρνταρόμπα τους.

Η πλήρης σύνδεσης της εφαρμογής βασίστηκε σε κεντρικές τεχνολογίες, όπως το React.js στο **Front-End** και του Node.js με Express.js στο **Back-End**, ενώ η **MongoDB** επιλέχθηκε ως η βάση δεδομένων. Το React.js διευκόλυνε τη δημιουργία μιας διαδραστικής και φιλικής

προς τον χρήστη διεπαφής, ανοίγοντας τον δρόμο στην επεκτασιμότητα και την εξέλιξη στον σχεδιασμό και τις λειτουργίες της εφαρμογής. Στο Back-End, το Node.js και το Express.js παρείχαν τη σταθερότητα και την ταχύτητα που απαιτούνται για την εξυπηρέτηση των αιτημάτων των χρηστών, την σωστή δομή των μοντέλων και την επικοινωνία με τη MongoDB, η οποία συνέβαλε στην αποτελεσματική διαχείριση των δεδομένων λόγω της ευέλικτης δομής της, προσφέροντας επεκτασιμότητα και ασφαλή αποθήκευση. Οι τεχνολογικές επιλογές, σε συνδυασμό με τον σχεδιασμό της εφαρμογής, συνέβαλαν στην επίτευξη των στόχων της αποδοτικότητας, επεκτασιμότητας και ασφάλειας που είχαμε θέσει.

Κατά την ανάπτυξη της εφαρμογής προέκυψαν διάφορα τεχνικά εμπόδια και προκλήσεις που απαιτούσαν άμεσες και πρακτικές λύσεις. Τις σημαντικότερες προκλήσεις αποτελούσαν η αποτελεσματική επικοινωνία μεταξύ Front-End και Back-End, η διασφάλιση της ασφάλειας των δεδομένων των χρηστών και η διατήρηση μιας απλής και ευχάριστης εμπειρίας χρήστη. Μέσα από ενδελεχή έλεγχο, συνεχείς δοκιμές και σταδιακή βελτίωση του κώδικα, αντιμετωπίστηκαν όλα τα τεχνικά και λειτουργικά εμπόδια που προέκυψαν. Η διαδικασία αυτή ενθάρρυνε τον σχεδιασμό της εφαρμογής με τέτοιο τρόπο ώστε να είναι ανοιχτή σε προσαρμογές και αναβαθμίσεις.

Η ανάπτυξη του Digital Wardrobe προσέφερε πολύτιμα μαθήματα και γνώσεις. Κατά τη διάρκεια της διαδικασίας, αναπτύξαμε μια βαθύτερη κατανόηση σχετικά με την συνεργασία και επικοινωνία του Front-End και Back-End, την σημασία της ασφαλούς διαχείρισης των βάσεων δεδομένων και την ανάγκη για μια καλά σχεδιασμένη αρχιτεκτονική που διευκολύνει τη συντήρηση και την επεκτασιμότητα. Επιπλέον, μέσα από την ολοκλήρωση της ανάπτυξης, δόθηκε ακόμη μεγαλύτερη αξία στην απλότητα και προσβασιμότητα στον σχεδιασμό της διεπαφής χρήστη, κάτι που κάνει την αλληλεπίδραση ευχάριστη και ενισχύει την εμπειρία των χρηστών.

Η εφαρμογή έχει δυνατότητες περαιτέρω ανάπτυξης και επέκτασης. Μελλοντικά, χάρη στην ήδη υπάρχουσα υποδομή, μπορούν εύκολα να προστεθούν νέες λειτουργίες, όπως η ενσωμάτωση τεχνολογιών τεχνητής νοημοσύνης για προσωποποιημένες προτάσεις συνόλων, βάσει των προτιμήσεων και των δημιουργημένων συνόλων των χρηστών. Επίσης, μπορούν να ενισχυθούν οι δυνατότητες οργάνωσης της ψηφιακής ντουλάπας, με επιπλέον φίλτρα και λειτουργίες κατηγοριοποίησης, ενώ η προσθήκη εργαλείων για προτάσεις νέων ειδών ένδυσης θα προσέφερε ακόμη μεγαλύτερη αξία στους χρήστες. Οι ιδέες αυτές συνδυάζουν την εξέλιξη με την παροχή προστιθέμενης αξίας, διασφαλίζοντας ότι η εφαρμογή θα μπορεί να συνεχίσει να ανταποκρίνεται στις αυξανόμενες ανάγκες των χρηστών και στις σύγχρονες τάσεις της μόδας και της τεχνολογίας.

## ΒΙΒΛΙΟΓΡΑΦΙΑ

- [1] Berlinger, M. (2022). *Why We Buy Clothes We Like And End Up With A Wardrobe We Hate*. MR PORTER The Journal.

<https://www.mrporter.com/en-us/journal/fashion/why-we-hate-our-clothes-full-wardrobe-nothing-to-wear-2022-17464068>

- [2] Stretton, S. (2020). *Why digital is more than just technology*. LinkedIn Pulse.

[https://www.linkedin.com/pulse/why-digital-more-than-just-technology-stephanie-stretton?trk=article-ssr-frontend-pulse\\_more-articles\\_related-content-card](https://www.linkedin.com/pulse/why-digital-more-than-just-technology-stephanie-stretton?trk=article-ssr-frontend-pulse_more-articles_related-content-card)

- [3] Jacobson, S. (2023). *Always Disorganised? Why Your Life is Such a Mess*. Harley Therapy: Mental Health Blog.

<https://www.harleytherapy.co.uk/counselling/disorganised.htm>

- [4] *4 Root Causes of a Messy Closet (And How to Counter Them)*. (2017). The Cleaning Advantage.

<https://www.thecleaningadvantage.com/4-root-causes-messy-clothes-closet-counter/>

- [5] *Ti είναι η ΔΕΙΤΥ;*. (2013). ADHD Hellas.

<https://www.adhdhellas.org/2013-09-13-13-14-13/ti-einai>

- [6] Magnus, W., Nazir, S., Anilkumar, A., C., & Shaban, K. (2023). *Attention Deficit Hyperactivity Disorder*. NCBI Bookshelf StatPearls.

<https://www.ncbi.nlm.nih.gov/books/NBK441838/>

- [7] Singh, A., C., Bhuvaneswari, M. (2023). *Does mental health limit organizational behavior, or not? A study drawn on resource conservation*. NCBI Bookshelf Frontiers in Psychology.

<https://pmc.ncbi.nlm.nih.gov/articles/PMC10399591/>

- [8] van der Kolk, B. (2000). *Posttraumatic stress disorder and the nature of trauma*. Taylor & Francis, 2(1), 7-22.

<https://www.tandfonline.com/doi/full/10.31887/DCNS.2000.2.1/bvdkolk#d1e155>

- [9] *Top 7 Ways Disorganized Closets Negatively Affect Your Life*. Organized Interiors Blog.

<https://www.organizedinteriors.com/blog/disorganized-closets/>

[10] *Mastering Wardrobe Challenges*. (2024). Private Service Alliance.

<https://privateservicealliance.com/mastering-wardrobe-challenges/>

[11] Beaird, J., Walker, A., & George, J. (2020). *The principles of beautiful web design*. SitePoint Pty Ltd.

[12] Jordana, A. (2024). *What Is JavaScript: A Beginner's Guide to the Basics of JS*. Hostinger Tutorials.

<https://www.hostinger.com/tutorials/what-is-javascript>

[13] Brown, E. (2019). *Web development with node and express: leveraging the JavaScript stack*. O'Reilly Media.

[14] Gillis, A., S. (2023). *What is MongoDB? Features and how it works*. Tech Target.

<https://www.techtarget.com/searchdatamanagement/definition/MongoDB>

[15] Karnik, N. (2018). *Introduction to Mongoose for MongoDB*. freeCodeCamp.

<https://www.freecodecamp.org/news/introduction-to-mongoose-for-mongodb-d2a7aa593c57/>

[16] Herbert, D. (2022). *What is React.js? Uses, Examples, & More*. Blog Hubspot.

<https://blog.hubspot.com/website/react-js>

[17] Hayes, A., Kindness, D. & Rathburn, D. (2024). *HyperText Markup Language (HTML): What It Is and How It Works*. Investopedia.

<https://www.investopedia.com/terms/h/html.asp>

[18] *What is CSS?* (2024). MDN Web Docs.

[https://developer.mozilla.org/en-US/docs/Learn/CSS/First\\_steps/What\\_is\\_CSS](https://developer.mozilla.org/en-US/docs/Learn/CSS/First_steps/What_is_CSS)

[19] Powell, T. (2010). *HTML & CSS: the complete reference*. McGraw-Hill, Inc..

[20] *Backend Development: Design and Implementation of Software for the Web*. (2016). George Mason University.

<https://cs.gmu.edu/~tlatoza/teaching/swe432f16/Lecture%2013%20-%20Backend%20Development.pdf>

[21] *Express Tutorial Part 3: Using a Database (with Mongoose)*. (2024). MDN Web Docs.

[https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express\\_Nodejs/mongoose](https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express_Nodejs/mongoose)

[22] Aziz, I. (2023). *Exploring How Websites Talk: A Beginner's Guide to HTTP Requests, APIs, and Backend Magic*. DEV Community.

<https://dev.to/ibrahzizo360/exploring-how-websites-talk-a-beginners-guide-to-http-requests-apis-and-backend-magic-4l48>

[23] Bojinov, V. (2018). *RESTful Web API Design with Node.js 10: Learn to create robust RESTful web services with Node.js, MongoDB, and Express.js*. Packt Publishing Ltd.

[24] Semah, B. (2022). *What Exactly is Node.js? Explained for Beginners*. freeCodeCamp.

<https://www.freecodecamp.org/news/what-is-node-js/>

[25] *Express/Node introduction*. (2024). MDN Web Docs.

[https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express\\_Nodejs/Introduction](https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express_Nodejs/Introduction)

[26] Bradshaw, S., Brazil, E., & Chodorow, K. (2019). *MongoDB: the definitive guide: powerful and scalable data storage*. O'Reilly Media.

<https://pepa.holla.cz/wp-content/uploads/2016/07/MongoDB-The-Definitive-Guide-2nd-Edition.pdf>

[27] Karnik, N. (2018). *Introduction to Mongoose for MongoDB*. freeCodeCamp.

<https://www.freecodecamp.org/news/introduction-to-mongoose-for-mongodb-d2a7aa593c57/>

[28] Rane, S. (2023). *How Do You Connect a Frontend Backend and Database?*. SourceBae.

<https://sourcebae.com/blog/how-do-you-connect-a-frontend-backend-and-database/>

[29] *JavaScript For Backend Development: An Introduction*. (2022). Blog Hubspot.

<https://blog.hubspot.com/website/java-backend>

[30] *JSON Web Token Introduction*. (2024). JWT.

<https://jwt.io/introduction>

[31] Weiss, M. (2014). *The Absolute Beginner's Guide to Node.js*. CloudBees.

<https://www.cloudbees.com/blog/node-js-tutorial>

[32] *HTTP Methods*. (2021). Contrive.

<https://www.contrive.mobi/aviorapi/HTTPMETHODS.html>

[33] *What is Express.js?*. Codecademy.

<https://www.codecademy.com/article/what-is-express-js>

[34] *Using middleware.* Express.js.

<https://expressjs.com/en/guide/using-middleware.html>

[35] *Connection Guide.* MongoDB Docs Menu.

<https://www.mongodb.com/docs/drivers/node/v3.6/fundamentals/connection/connect/>

[36] *Multer Middleware.* Express.js.

<https://expressjs.com/en/resources/middleware/multer.html>

[37] *What Is a JWT & How It Works.* (2024). descope.

<https://www.descope.com/learn/post/jwt>

[38] *bcrypt.js.* (2016). npm.js.

<https://www.npmjs.com/package/bcryptjs>

[39] Charity, D., T. (2024). *How to Hash Passwords with bcrypt in Node.js.* freeCodeCamp.

<https://www.freecodecamp.org/news/how-to-hash-passwords-with-bcrypt-in-nodejs/>

[40] Zanini, A. (2023). *Using Helmet in Node.js to secure your application.* LogRocket Blog.

<https://blog.logrocket.com/using-helmet-node-js-secure-application/#what-helmet>

[41] Seliesh, J. (2023). *Securing Your Node.js Application Security with Helmet.* Medium.

<https://medium.com/@selieshjksofficial/securing-your-node-js-application-security-with-helmet-296377385d07>

[42] Sentre, T. (2023). *How to Use Morgan in Your Node.js Project.* DEV Community.

<https://dev.to/devland/how-to-use-morgan-in-your-nodejs-project-21im>

[43] NamanCourses. (2024). *What is MORGAN in Node.js ?.* GeeksforGeeks.

<https://www.geeksforgeeks.org/what-is-morgan-in-node-js/>

[44] Singhal, A. (2023). *CORS in Nodejs.* Scaler Topics.

<https://www.scaler.com/topics/nodejs/cors-in-node-js/>

[45] Saragam. (2023). *CORS in Node.js with Express.* Medium

<https://saragam.medium.com/cors-in-node-js-with-express-99b355e93def>

[46] Gourav, H. (2024). *Express express.Router() Function*. GeeksforGeeks.

<https://www.geeksforgeeks.org/express-js-express-router-function/>

[47] Kumar, R. (2023). *Understanding Express.js Router (Node.js): A Comprehensive Guide*. DEV Community

<https://dev.to/mrrishimeena/understanding-expressjs-router-nodejs-a-comprehensive-guide-59o8>

[48] Stork, V. (2021). *How to Use Node Environment Variables with a DotEnv File for Node.js and npm*. freeCodeCamp.

<https://www.freecodecamp.org/news/how-to-use-node-environment-variables-with-a-dotenv-file-for-node-js-and-npm/>

[49] Shrestha, M. (2023). *What is dotenv in NodeJS?*. LinkedIn Pulse.

<https://www.linkedin.com/pulse/what-dotenv-nodejs-manoj-shrestha>

[50] Saharawat, V. (2024). *Bodyparser: Middleware In Node.js*. PW Skills.

<https://pwskills.com/blog/bodyparser-middleware-in-node-js/>

[51] Sharma, A. (2024). *A Complete And Comprehensive Guide To Understand Body Parser In Express JS*. Simplilearn.

<https://www.simplilearn.com/tutorials/nodejs-tutorial/body-parser-in-express-js>

[52] Nguyen, C. (2023). *Using express-validator for data validation in NodeJS*. Medium.

<https://howtodevez.medium.com/using-express-validator-for-data-validation-in-nodejs-6946af d9d67e>

[53] Singh, C. & Selvan, A. (2024). *How to Handle Form Inputs Efficiently with Express-Validator in ExpressJs*. DigitalOcean.

<https://www.digitalocean.com/community/tutorials/how-to-handle-form-inputs-efficiently-with-express-validator-in-express-js>

[54] Dwivedi, A. (2023). *Path Module in Node JS*. Scaler Topics.

<https://www.scaler.com/topics/nodejs/path-module-in-node-js/>

[55] Hooda, P. (2024). *Async Await in Node.js*. GeeksforGeeks.

<https://www.geeksforgeeks.org/async-await-in-node-js/>

- [56] Godbolt, M. (2016). *Frontend architecture for design systems: a modern blueprint for scalable and sustainable websites.* " O'Reilly Media, Inc.".
- [57] Vipul, A. M., & Sonpatki, P. (2016). *ReactJS by Example-Building Modern Web Applications with React*. Packt Publishing Ltd.
- [58] Mwaura, W. (2024). *Making HTTP requests with Axios*. CircleCI Blog.  
<https://circleci.com/blog/making-http-requests-with-axios/>
- [59] Governor, J., Hinchcliffe, D., & Nickull, D. (2009). *Web 2.0 Architectures: What entrepreneurs and information architects need to know.* " O'Reilly Media, Inc.".
- [60] Herbert, D. (2022). *What is React.js? Uses, Examples, & More*. Blog Hubspot.  
<https://blog.hubspot.com/website/react-js>
- [61] Hutsulyak, O. (2024). *10 Key Reasons Why You Should Use React for Web Development*. TechMagic.  
<https://www.techmagic.co/blog/why-we-use-react-js-in-the-development>
- [62] Johnson, J. (2020). *Designing with the mind in mind: simple guide to understanding user interface design guidelines*. Morgan Kaufmann.
- [63] O'Grady, B. (2024). *What is HTML? An Introduction*. Code Institute.  
<https://codeinstitute.net/global/blog/what-is-html-and-why-should-i-learn-it/>
- [64] Jha, S. (2023). *What Is JSX and why use it in ReactJS*. Medium.  
<https://medium.com/@shivamjha2436/what-is-javascript-an-why-use-it-in-reactjs-f3a8890e6d8e>
- [65] Fedosejev, A. (2015). *React.js essentials*. Packt Publishing Ltd.
- [66] Thorndyke, K. (2021). *What Is CSS Used For?*. Codeacademy.  
<https://www.codecademy.com/resources/blog/what-is-css-used-for/>
- [67] Ullman, L. (2012). *Modern JavaScript: Develop and Design*. Peachpit Press.
- [68] Islam Naim, N. (2017). *ReactJS: An Open Source JavaScript Library for Front-end Development*. Theseus.  
<https://www.theseus.fi/handle/10024/130495>
- [69] Cheston, R. (2024). *A Beginner's Guide to Using Axios in Node.js: Simplifying HTTP Requests*. Medium.

[https://medium.com/@reggiechesteron/a-beginners-guide-to-using-axios-in-node-js-simplifying  
-http-requests-441291fef064](https://medium.com/@reggiechesteron/a-beginners-guide-to-using-axios-in-node-js-simplifying-http-requests-441291fef064)

[70] Ijaz, U. (2023). *A brief introduction to Axios*. Rapid API Guides.

<https://rapidapi.com/guides/what-is-axios>

[71] Sengupta, D., Singhal, M., & Corvalan, D. (2016). *Getting Started with React*. Packt Publishing Ltd.

[72] Abajyan, A. (2018). *Web development: frontend using CSS with modern techniques*. AUA Institutional Repository.

<https://dspace.aua.am/xmlui/handle/123456789/214>

