



ΙΟΝΙΟ ΠΑΝΕΠΙΣΤΗΜΙΟ
Τμήμα Πληροφορικής

ΕΞΥΠΝΕΣ ΤΕΧΝΟΛΟΓΙΕΣ ΚΑΙ ΕΦΑΡΜΟΓΕΣ

Ακαδ. Έτος: 2020-2021

Εργασία Εξαμήνου

Σχεδιασμός & Ανάπτυξη Έξυπνης Εφαρμογής “Beehive Control”

Μέλη Ομάδας

Αριθμός μητρώου

Π2017131

Π2017005

Π2017082

Ονοματεπώνυμο

Γιώργος Μελιδονιώτης

Γιώργος Γεραρχάκης

Πασχάλης Γρίβας

E-mail

p17meli@ionio.gr

p17gera@ionio.gr

p17griv@ionio.gr

Περιεχόμενα

Περιεχόμενα	2
1. Εισαγωγή	3
1.1 Το Πρόβλημα	3
1.2 Προτεινόμενη Λύση – Εφαρμογή	4
1.2.1 Χρήστες της Εφαρμογής	6
2. Μεθοδολογία	7
2.1 Αρχιτεκτονική της Εφαρμογής	7
2.2 Μετασχηματισμός Δεδομένων	8
2.3 Μεθοδολογία Ανάπτυξης ενός Demo της Εφαρμογής	9
2.3.1 Τεχνολογικοί Περιορισμοί	9
2.3.2 Περιγραφή της Υλοποίησης	10
3. Αξιολόγηση - Συμπεράσματα	19
3.1 Τρόπος Αξιολόγησης της Εφαρμογής	19
3.2 Τεχνολογικοί και Άλλοι Περιορισμοί	19
3.3 Συμπεράσματα και Προοπτικές Μελλοντικής Αξιοποίησης	20
Αναφορές – Πηγές	21

Κεφάλαιο 1

Εισαγωγή

1.1 Το Πρόβλημα

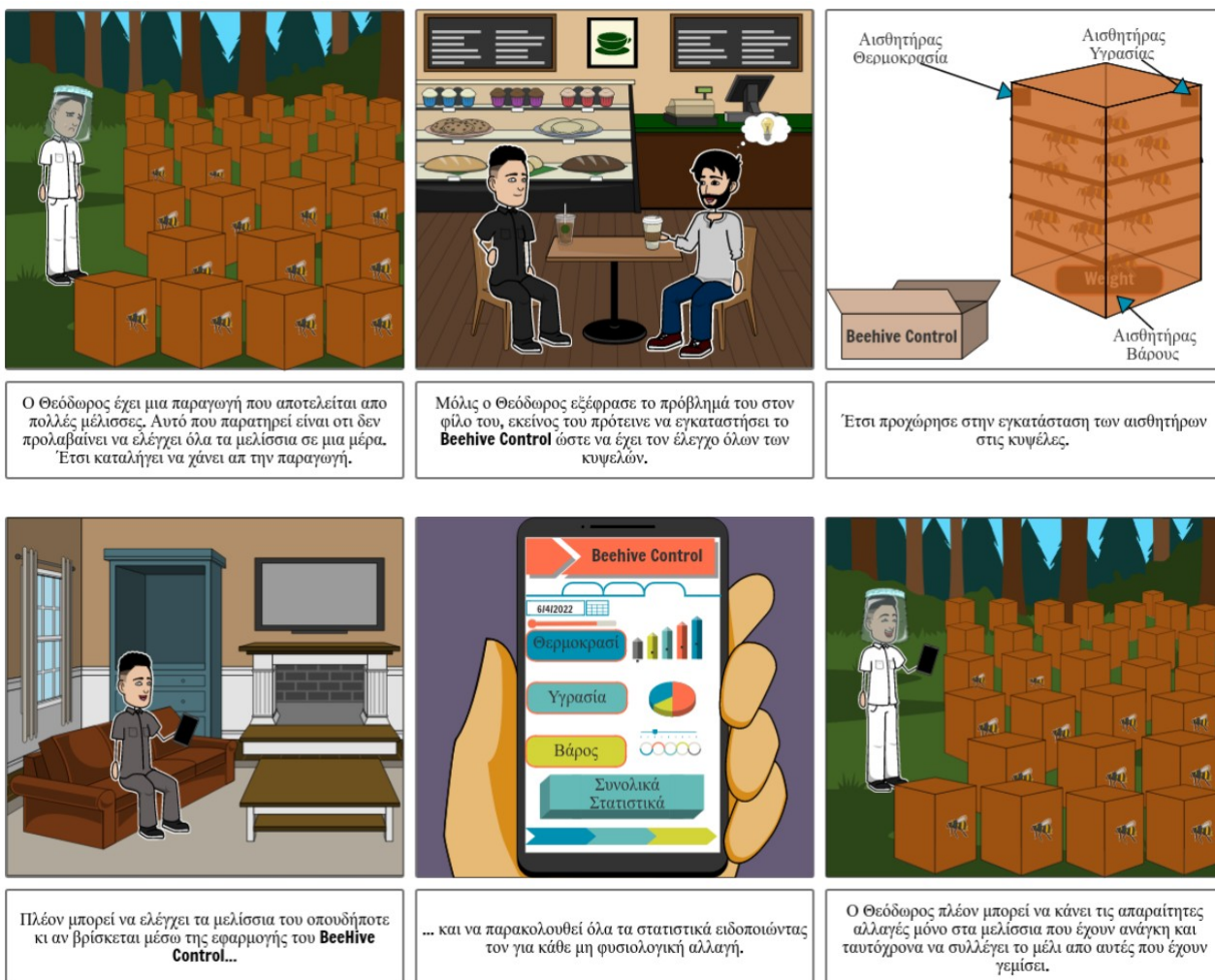
Η μελισσοκομεία είναι ένα απαιτητικό επάγγελμα και οποιουδήποτε είδους αποτελεσματικό τεχνολογικό βοήθημα (καινοτομία) ή λύση πληροφορικής, μπορεί να προσφέρει σημαντικά και να κριθεί απαραίτητη για τον επαγγελματία μελισσοκόμο. Ανάμεσα στις εργασίες, τις οποίες πρέπει να πραγματοποιεί ένας μελισσοκόμος, είναι η εκτροφή των μελισσών, η συγκομιδή και πώληση του μελιού και των άλλων παραγόμενων προϊόντων, της μέλισσας. Επιπλέον, δουλειά του επαγγελματία, είναι η συντήρηση και η επισκευή των κυψελών, η αύξηση της χωρητικότητάς τους και η τοποθέτηση καινούριων, στις περιπτώσεις κατά τις οποίες ο πληθυσμός μελισσών αυξηθεί σημαντικά.

Η συλλογή του μελιού θα πρέπει να γίνεται σε τέτοιες ποσότητες, ώστε να μένει τροφή για το σμήνος. Επίσης, συχνά είναι αναγκαίο να προστίθεται επιπλέον τροφή στις κυψέλες, προκειμένου να συντηρούνται οι μέλισσες, ιδίως την περίοδο του χειμώνα. Επομένως, ο μελισσοκόμος θα πρέπει να επισκέπτεται σε τακτά χρονικά διαστήματα τις κυψέλες του, ώστε να ελέγχει τα επίπεδα πληρότητας τους. Ωστόσο, η διαδικασία αυτή πολλές φορές εμπεριέχει κάποια προβλήματα.

Δεδομένου ότι συχνά οι μελισσοκόμοι μετακινούν τις κυψέλες τους, περισσότερες από μία φορές το χρόνο, σε διάφορες τοποθεσίες (θέλοντας να παράξουν διαφορετική ποικιλία μελιού ή να προφυλάξουν τις μέλισσες από δυσμενείς καιρικές συνθήκες), είναι αναγκασμένοι να πραγματοποιούν μεγάλα δρομολόγια, προκειμένου να ελέγχουν τα επίπεδα των κυψελών. Αυτό σε πρώτο επίπεδο, έχει ως αποτέλεσμα την επιδείνωση της καθημερινότητας των μελισσοκόμων ενώ μακροπρόθεσμα την αύξηση του κόστους παραγωγής. Επίσης, στην περίπτωση κατά την οποία ένας μελισσοκόμος διαχειρίζεται έναν μεγάλο αριθμό κυψελών, προκύπτει μία ακόμα δυσκολία στην διαδικασία αυτή. Στην πράξη, ο μελισσοκόμος δεν προλαβαίνει μέσα στην μέρα να ελέγξει όλες τις κυψέλες του, με αποτέλεσμα να δημιουργούνται διάφορα προβλήματα στο σμήνος.

Επίσης, πέραν της αλλαγής της τοποθεσίας των κυψελών, κρίνεται απαραίτητη και η συχνή επιτόπια μετακίνησή τους. Η ανάγκη αυτή, προκαλείται λόγω διάφορων γεγονότων, τα οποία

μπορεί να συμβούν, κυρίως σχετιζόμενα με τις καιρικές συνθήκες και άλλους φυσικούς παράγοντες στο περιβάλλον των κυψελών. Για παράδειγμα, μία κυψέλη είναι πιθανό να εκτίθεται πολλές ώρες στον ήλιο και να αναπτύσσει εσωτερικά υψηλή θερμοκρασία, αποκλίνοντας από τις ιδανικές συνθήκες διαβίωσης των μελισσών. Επομένως, προκειμένου να εντοπίζονται αυτά τα φαινόμενα, ο μελισσοκόμος θα πρέπει να επισκέπτεται τις κυψέλες του αρκετές φορές μέσα στην ημέρα.



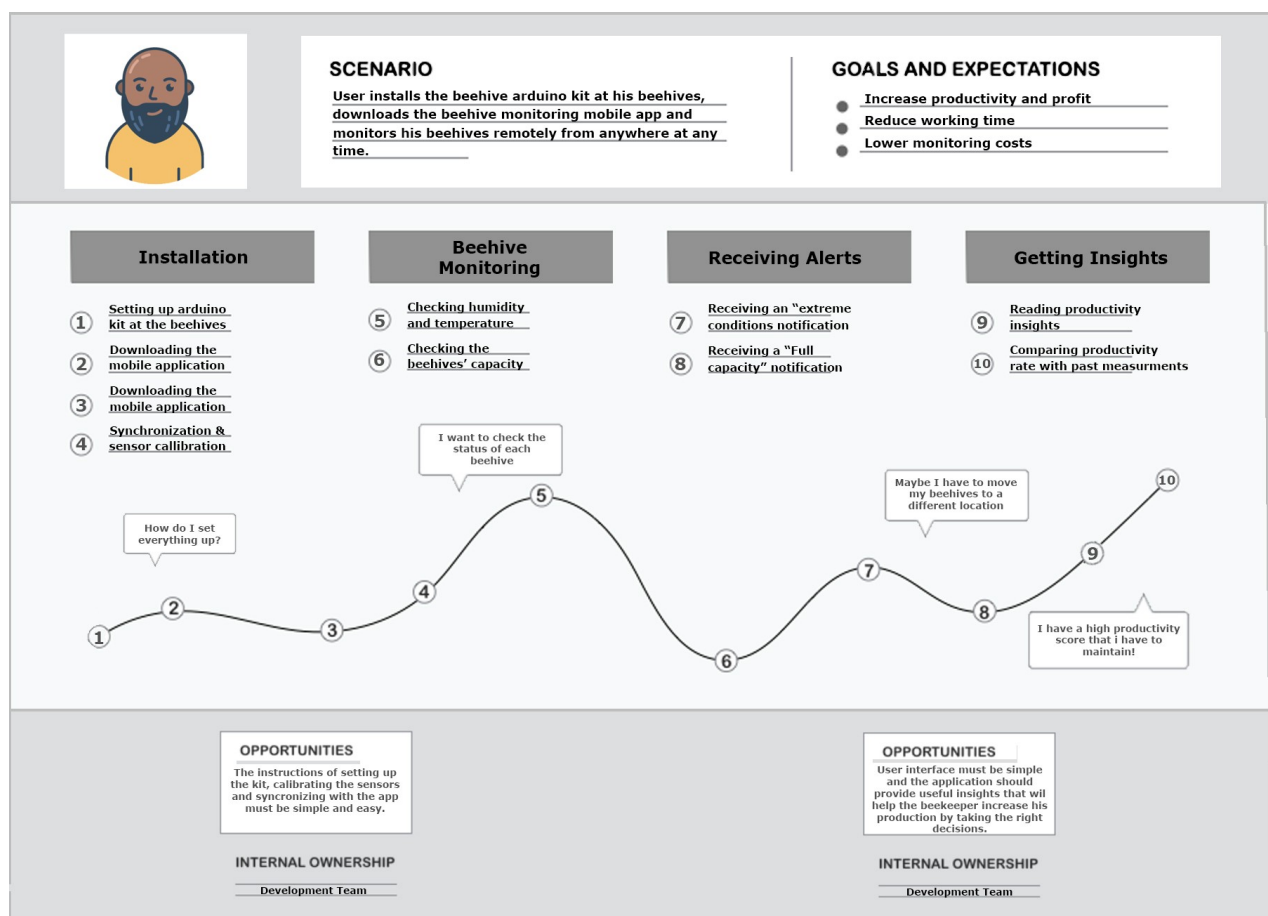
Εικόνα 1: Storyboard - Ένα από τα προβλήματα που καλούνται να αντιμετωπίσουν οι μελισσοκόμοι σχετικά με τον έλεγχο των κυψελών.

1.2 Προτεινόμενη Λύση - Εφαρμογή

Προκειμένου να δοθεί μία λύση στα προαναφερθέντα προβλήματα, τα οποία καλούνται να αντιμετωπίσουν οι μελισσοκόμοι, προτείνεται η εφαρμογή “Beehive Control”. Πρόκειται για ένα πακέτο κατάλληλου υλικού εξοπλισμού και λογισμικού, το οποίο επιτρέπει την εξ αποστάσεως λήψη βασικών πληροφοριών για κάθε μία κυψέλη, όπως η εσωτερική θερμοκρασία και υγρασία της και το ποσοστό πληρότητας της. Πιο συγκεκριμένα, δίνεται στον χρήστη η δυνατότητα να αγοράσει και να εγκαταστήσει τις ειδικά διαμορφωμένες, με τους κατάλληλους αισθητήρες και εξαρτήματα,

πλακέτες Arduino “Arduino Kit”, στις κυψέλες τους. Έπειτα, μέσω της web εφαρμογής ή της εφαρμογής για φορητές συσκευές Android, ο χρήστης μπορεί να επιλέξει τα Kit του και να παρακολουθεί, σε πραγματικό χρόνο, από οποιαδήποτε τοποθεσία, την πληρότητα και τις συνθήκες των κυψελών του. Επίσης, σε περίπτωση που εντοπιστούν τιμές, οι οποίες θεωρούνται εκτός των φυσιολογικών ορίων, οι εφαρμογές ενημερώνουν τον χρήστη, μέσω κατάλληλων ειδοποιήσεων, έτσι ώστε να προβεί σε διορθωτικές ενέργειες.

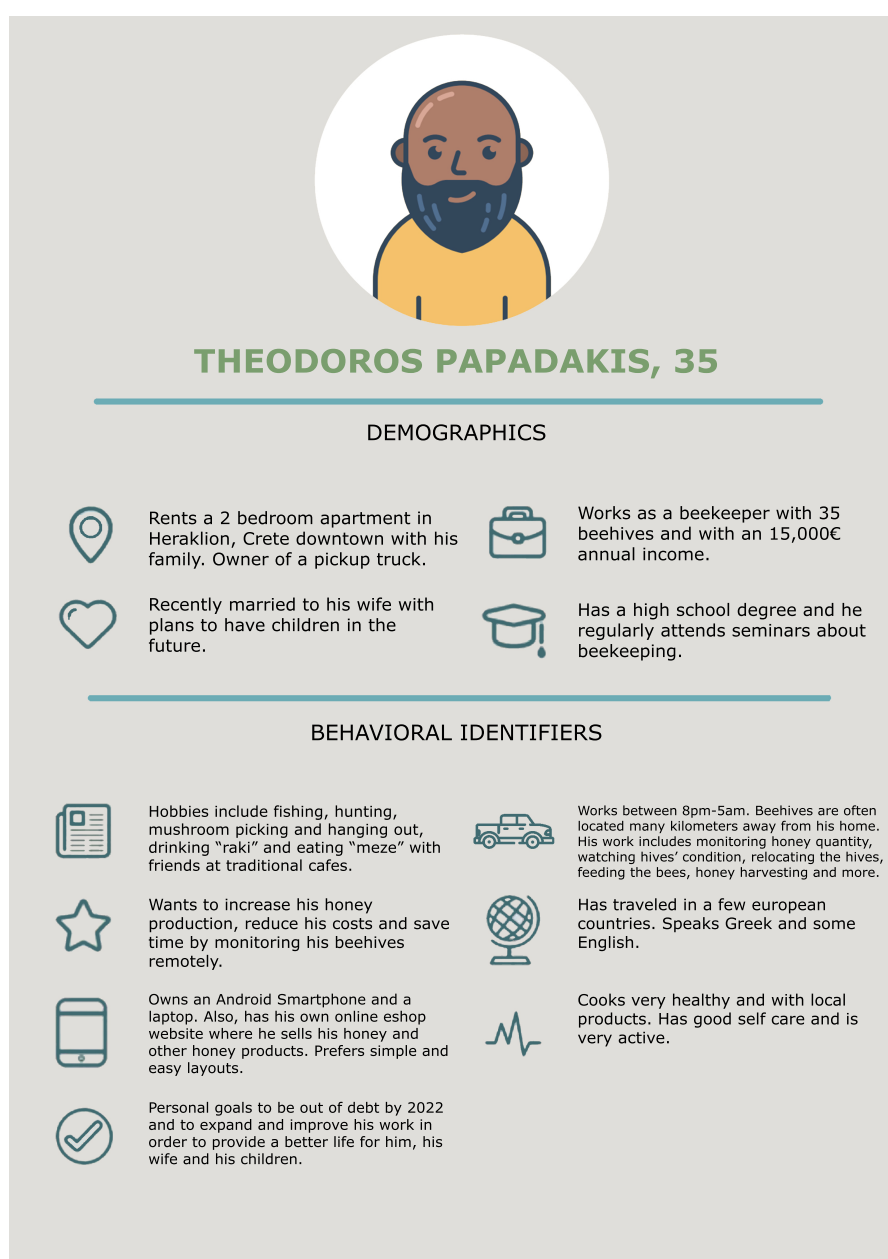
Εφαρμόζοντας την προτεινόμενη αυτή λύση, ο μελισσοκόμος απαλλάσσεται σε μεγάλο βαθμό από τις δυσκολίες επιτήρησης και ελέγχου των κυψελών. Σε πρακτικό επίπεδο, ο έλεγχος γίνεται μέσα σε λίγα λεπτά και πλέον δεν υπάρχουν οι περιπτώσεις όπου ο μελισσοκόμος δεν προλάβαινε μέσα στη μέρα να ελέγξει όλες τις κυψέλες του. Επίσης, δεδομένου ότι ειδοποιείται για τα επίπεδα πληρότητας, πλέον ο μελισσοκόμος μπορεί να επεμβαίνει μόνο σε ένα μέρος των κυψελών, το οποίο απαιτεί κάποια ενέργεια από μέρος του, εξοικονομώντας επιπλέον χρόνο. Τέλος, λόγω του γεγονότος ότι ο χρήστης λαμβάνει ειδοποιήσεις και για ακραίες τιμές στις συνθήκες των κυψελών, δεν απαιτείται να πραγματοποιεί συχνά δρομολόγια ώστε να εντοπίζει τέτοια περιστατικά. Αυτό, έχει ως αποτέλεσμα την εξοικονόμηση χρόνου και μία σημαντική μείωση κόστος παραγωγής, το οποίο εν τέλει ισοδυναμεί με την αύξηση του εισοδήματος του μελισσοκόμου.



Εικόνα 2: User Journey Map - Η εγκατάσταση και οι λειτουργίες της εφαρμογής μέσω ενός σεναρίου.





1.2.1 Χρήστες της Εφαρμογής

Οι κατηγορία χρηστών, στην οποία απευθύνεται η εφαρμογή είναι αρκετά συγκεκριμένη. Πρόκειται για επαγγελματίες μελισσοκόμους, με μία σχετικά χαμηλού επιπέδου εξοικείωση χρήσης web ή κινητών εφαρμογών, οι οποίοι επιθυμούν να βελτιώσουν - αυτοματοποιήσουν την διαδικασία ελέγχου των κυψελών τους, να αυξήσουν το εισόδημά τους, μειώνοντας τα έξοδά τους, και να εκσυγχρονίσουν τις εγκαταστάσεις τους. Επίσης, οι χρήστες θα πρέπει να είναι σε θέση να ακολουθήσουν τις οδηγίες, ώστε να εγκαταστήσουν και να προσαρμόσουν τον εξοπλισμό στις κυψέλες τους.










THEODOROS PAPADAKIS, 35

DEMOGRAPHICS

	Rents a 2 bedroom apartment in Heraklion, Crete downtown with his family. Owner of a pickup truck.		Works as a beekeeper with 35 beehives and with an 15,000€ annual income.
	Recently married to his wife with plans to have children in the future.		Has a high school degree and he regularly attends seminars about beekeeping.

BEHAVIORAL IDENTIFIERS

	Hobbies include fishing, hunting, mushroom picking and hanging out, drinking "raki" and eating "meze" with friends at traditional cafes.		Works between 8pm-5am. Beehives are often located many kilometers away from his home. His work includes monitoring honey quantity, watching hives' condition, relocating the hives, feeding the bees, honey harvesting and more.
	Wants to increase his honey production, reduce his costs and save time by monitoring his beehives remotely.		Has traveled in a few european countries. Speaks Greek and some English.
	Owens an Android Smartphone and a laptop. Also, has his own online eshop website where he sells his honey and other honey products. Prefers simple and easy layouts.		Cooks very healthy and with local products. Has good self care and is very active.
	Personal goals to be out of debt by 2022 and to expand and improve his work in order to provide a better life for him, his wife and his children.		

Εικόνα 3: User Persona – Παράδειγμα ενός υποψήφιου χρήστη της προτεινόμενης εφαρμογής.

Κεφάλαιο 2

Μεθοδολογία

2.1 Αρχιτεκτονική της Εφαρμογής

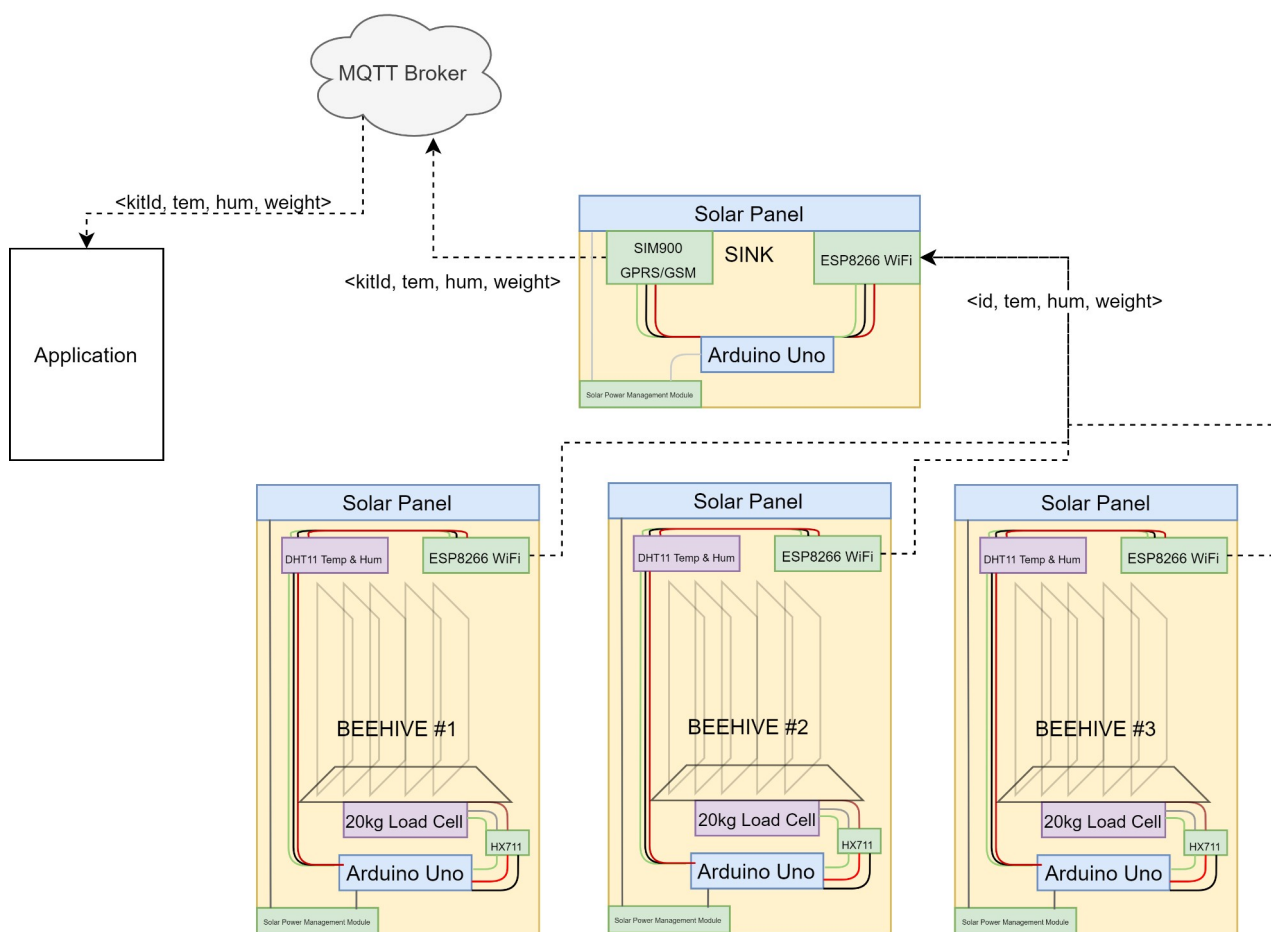
Αρχικά, σε κάθε κυψέλη, χρησιμοποιείται μία πλακέτα μικροελεγκτή [Arduino Uno](#). Μέσω των ψηφιακών και αναλογικών της υποδοχών συνδέεται ο αισθητήρας μέτρησης θερμοκρασίας και υγρασίας [DHT-11](#), ο οποίος τοποθετείται σε κάποιο σημείο στο εσωτερικό της κυψέλης. Ο συγκεκριμένος αισθητήρας ανιχνεύει τιμές σχετικής υγρασίας (RH) από 20% έως 90%, με ακρίβεια 5%, ενώ οι τιμές της θερμοκρασίας, οι οποίες ανιχνεύονται, είναι από 0°C έως 50°C, με ακρίβεια 2°C.

Προκειμένου να ελέγχεται πληρότητα της κάθε κυψέλης, είναι απαραίτητη η τοποθέτηση μίας επιφάνειας-βάσης (ξύλινη ή μεταλλική) κάτω από τα τελάρα-πλαίσια της κυψέλης. Κάτω από την επιφάνεια αυτή, τοποθετείται ένας αισθητήρας μέτρησης βάρους [20kg Load Cell](#) (υψηλής ακρίβειας και μέγιστου βάρους τα 20kg), ο οποίος συνδέεται με τον μικροελεγκτή Arduino της κυψέλης μέσω των αναλογικών υποδοχών του. Στην σύνδεση αυτή, είναι απαραίτητη η μεσολάβηση μίας πλακέτας HX711, η οποία λειτουργεί ως ενισχυτής των σημάτων του αισθητήρα βάρους.

Επίσης, πέρα από τα Arduino, τα οποία είναι τοποθετημένα σε κάθε κυψέλη, είναι απαραίτητος ένας ακόμα μικροελεγκτής, ο οποίος θα αναλαμβάνει την συλλογή των δεδομένων από όλους τους μικροελεγκτές των κυψελών (“SINK”). Προκειμένου όλα τα Arduino να στέλνουν ασύρματα τις τιμές των αισθητήρων τους σε αυτόν τον κόμβο, είναι απαραίτητο όλοι οι μικροελεγκτές να διαθέτουν ένα WiFi module [ESP8266](#). Επίσης, ο κόμβος “SINK” είναι υπεύθυνος για την αποστολή των δεδομένων στον κεντρικό MQTT Broker, ο οποίος εκτελείται σε κάποιον απομακρυσμένο εξυπηρετητή. Για να γίνει αυτό, ο κόμβος διαθέτει ένα επιπλέον module, το [SIM900](#), προκειμένου να επικοινωνεί με τον Broker μέσω του δικτύου κινητής τηλεφωνίας (GSM).

Τέλος, για να είναι ο κάθε κόμβος (κυψέλες & “SINK”) ενεργειακά αυτόνομος, είναι δυνατή η τροφοδότηση των μικροελεγκτών μέσω [ηλιακών συλλεκτών](#). Τα ηλιακά αυτά πάνελ, τοποθετούνται στο πάνω μέρος κάθε κυψέλης και συνδέονται με ένα module διαχείρισης ηλιακής ενέργειας (Solar

Power Management Module). Το module αυτό, μετατρέπει την ενέργεια σε κατάλληλη μορφή και φορτίζει μία επαναφορτιζόμενη μπαταρία λιθίου [14500](#) 3.7V. Έπειτα, το module αυτό τροφοδοτεί την πλακέτα Arduino, μέσω της θύρας USB.



Εικόνα 4: Η αρχιτεκτονική της εφαρμογής.

2.2 Μετασχηματισμός Δεδομένων

Όπως αναφέρθηκε στην προηγούμενη ενότητα, ο κόμβος “SINK” συλλέγει τα δεδομένα των κυψελών. Πιο συγκεκριμένα, δέχεται από κάθε μικροελεγκτή ένα αλφαριθμητικό, το οποίο περιλαμβάνει ένα αναγνωριστικό της κυψέλης (“id”) και τις τιμές των αισθητήρων θερμοκρασίας και υγρασίας (“tem” & “hum”) και βάρους (“weight”), εκείνη τη χρονική στιγμή. Στη συνέχεια, ο κόμβος προσθέτει στο αλφαριθμητικό αυτό τον μοναδικό αναγνωριστικό κωδικό (“kitId”) του τρέχοντος Kit και το στέλνει στον Broker. Στην πράξη, πραγματοποιεί ένα POST request μέσω του πρωτοκόλλου HTTP στον εξυπηρετητή, εκείνος απομονώνει τον αναγνωριστικό κωδικό του Kit και στη συνέχεια στέλνει (publish) το αλφαριθμητικό στο αντίστοιχο topic, στον MQTT Broker, ο οποίος εκτελείται στον ίδιο εξυπηρετητή.

Οι εφαρμογές του χρήστη λαμβάνουν (subscribe στο αντίστοιχο kitId-topic) τα δεδομένα (αλφαριθμητικό) και απομονώνουν (parse) τις τιμές του κάθε αισθητήρα, της κάθε κυψέλης, ώστε σε πρώτο επίπεδο να προβληθούν με κατάλληλο τρόπο στη διεπαφή της εφαρμογής. Επιπλέον, με κάθε αλφαριθμητικό, το οποίο λαμβάνεται, γίνεται έλεγχος των τιμών ώστε να διαπιστωθεί εάν αυτές βρίσκονται εκτός κάποιων συγκεκριμένων ορίων, προκειμένου να ειδοποιηθεί ο χρήστης. Επίσης, αποθηκεύοντας τις τιμές σε μία βάση δεδομένων, μπορούν να υπολογίζονται και να προβάλλονται οι μέσοι όροι της θερμοκρασίας και της υγρασίας, για κάθε κυψέλη, ο δείκτης παραγωγικότητας κάθε μίας (ρυθμός με τον οποίο γεμίζουν – αυξάνεται το βάρος τους), ενώ δίνεται η δυνατότητα προβολής αυτών των δεδομένων, σε συνάρτηση με τον χρόνο, μέσω διαγραμμάτων.

2.3 Μεθοδολογία Ανάπτυξης ενός Demo της Εφαρμογής

Η εφαρμογή χρήστη, στο τρέχον στάδιο ανάπτυξής της (demo) την δεδομένη χρονική στιγμή συγγραφής της παρούσας αναφοράς, λαμβάνει τα δεδομένα από τον Broker, στην συνέχεια εμφανίζει τις τιμές των αισθητήρων της κάθε κυψέλης, ελέγχει τις τιμές και προειδοποιεί τον χρήστη, μέσω κατάλληλων μηνυμάτων, στις περιπτώσεις όπου αυτές βρίσκονται εκτός κάποιων ορίων.

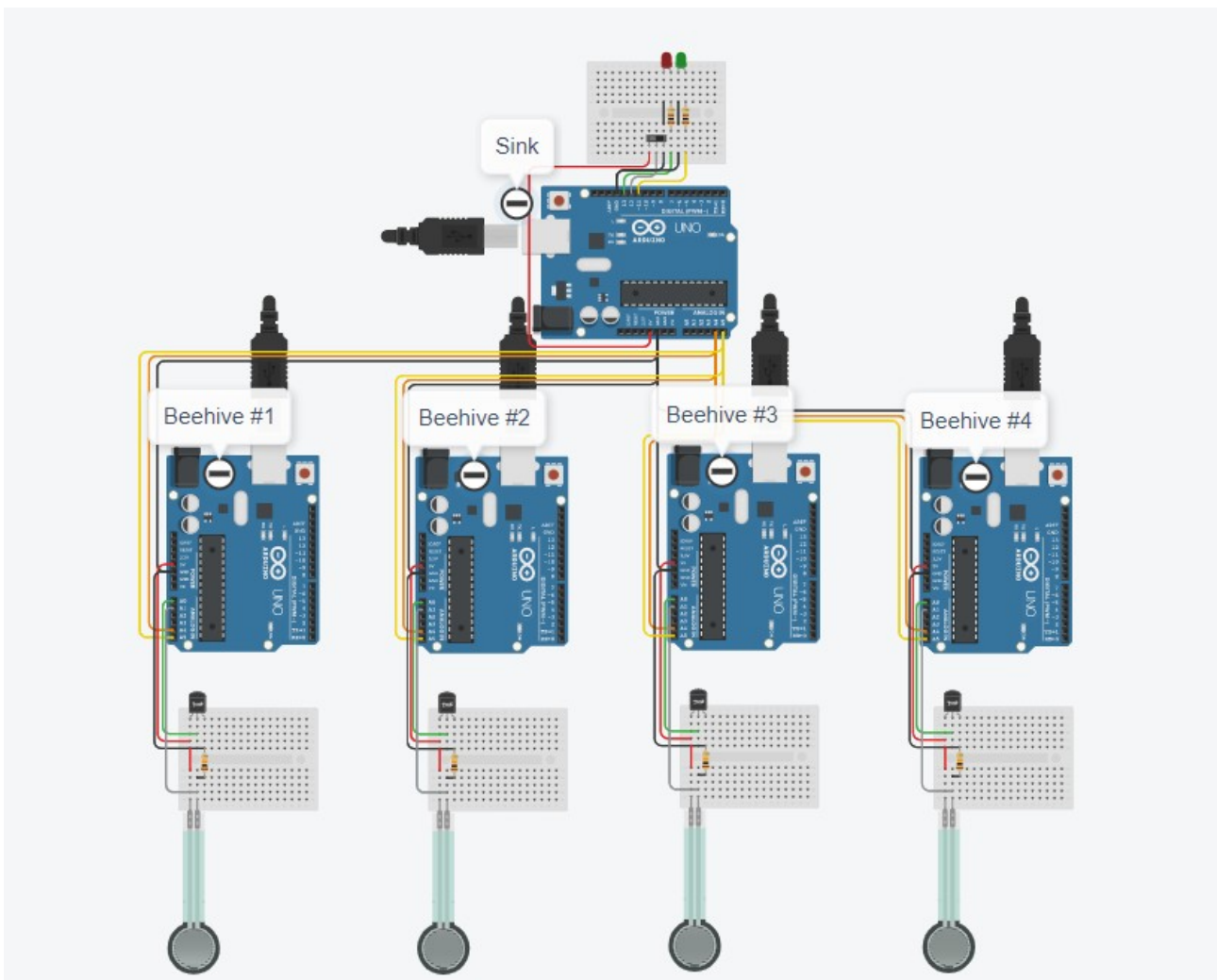
2.3.1 Τεχνολογικοί Περιορισμοί

Προκειμένου να προσομοιωθεί ο βασικός υλικός εξοπλισμός της εφαρμογής, ο οποίος περιγράφηκε σε προηγούμενη ενότητα, και οι συνθήκες ενός περιβάλλοντος, αντίστοιχο με ένα στο οποίο θα βρίσκονταν οι κυψέλες ενός μελισσοκόμου, χρησιμοποιήθηκε το online εργαλείο μοντελοποίησης και σχεδιασμού [Tinkercad](#). Ωστόσο, το εργαλείο παρέχει ένα περιορισμένο αριθμό και συγκεκριμένα είδη αισθητήρων, εξαρτημάτων και μικροελεγκτών. Επομένως, για τον λόγο αυτό δεν είναι δυνατό να υλοποιηθεί επακριβώς η αρχιτεκτονική της εφαρμογής, όπως περιγράφηκε στην ενότητα 2.1.

Πιο συγκεκριμένα, για την μέτρηση της θερμοκρασίας, χρησιμοποιήθηκε ο αισθητήρας [TMP36](#), ενώ δεν παρέχεται αισθητήρας μέτρησης υγρασίας. Επίσης, σε ότι αφορά την προσομοίωση μέτρησης του βάρους, χρησιμοποιήθηκε ο αισθητήρας Force Sensor (Force Sensing Resistor – [FSR](#)), οι τιμές του οποίου μετατράπηκαν ώστε να είναι πιο κοντά στις τιμές, τις οποίες θα έδινε ένας αισθητήρας Load Sensor. Τέλος, δεδομένου ότι το εργαλείο δεν υποστηρίζει την “ασύρματη” επικοινωνία μεταξύ των διάφορων μικροελεγκτών, μέσω κάποιου εικονικού WiFi module, η προσομοίωση της επικοινωνίας των συσκευών έγινε χρησιμοποιώντας ενσύρματη σύνδεση και το πρωτόκολλο επικοινωνίας I2C.

2.3.2 Περιγραφή της Υλοποίησης

Όπως αναφέρθηκε στη προηγούμενη ενότητα, για τον σχεδιασμό και προσομοίωση του εξοπλισμού, δηλαδή ενός “Beehive Control Arduino Kit”, χρησιμοποιήθηκε το εργαλείο Tinkercad. Αναλυτικότερα, χρησιμοποιήθηκαν τέσσερα Arduino Uno R3 boards, στις αναλογικές υποδοχές των οποίων συνδέθηκαν ένας αισθητήρας θερμοκρασίας και ένας αισθητήρας δύναμης (Force Sensor). Το κάθε ένα από αυτά αναπαριστά μία κυψέλη και είναι ενσύρματα συνδεδεμένο με ένα κεντρικό πέμπτο board, το οποίο λειτουργεί ως ο κόμβος “SINK”, δηλαδή συλλέγει και “στέλνει” τα δεδομένα.



Εικόνα 5: Αναπαράσταση του “Beehive Control Arduino Kit” στο Tinkercad.

Σε ότι αφορά τον τρόπο με τον οποίο λειτουργούν τα Arduino, τα οποία αντιπροσωπεύουν τις κυψέλες, ο κώδικας που εκτελείται είναι ο ίδιος. Πιο συγκεκριμένα, μέσω της συνάρτησης “onRequest” της βιβλιοθήκης “Wire.h”, η οποία αποτελεί μία υλοποίηση του πρωτοκόλλου I2C, ελέγχεται εάν ο κόμβος “SINK” έχει ζητήσει να του σταλούν δεδομένα. Σε περίπτωση που ληφθεί

ένα τέτοιο αίτημα, το τρέχον Arduino στέλνει τις τιμές των αισθητήρων την τρέχουσα χρονική στιγμή.

```
#include <Wire.h>
...
// Function that executes whenever data is requested by master
void requestEvent()
{
    // Getting the voltage reading from the temperature sensor
    int reading = analogRead(TEMP);
    ...
    // Getting the voltage reading from the force sensor
    fsrReading = analogRead(WEIGHT);
    ...
    String stringOne = String(temperatureC, 1);
    String stringTwo = String(netWeight, 1);
    String message = String("1:" + stringOne + ":" + stringTwo);
    char buf[15];
    message.toCharArray(buf, 15);
    Wire.write(buf);
}
```

Απόσπασμα_Κώδικα 1: Λήψη αιτήματος από τον “SINK” αποστολής και αποστολή των τιμών των αισθητήρων της τρέχουσας κυψέλης. - Αρχείο: “beehive_control_kit_beehive.ino”.

Από την άλλη μεριά, το Arduino, το οποίο αντιπροσωπεύει τον κόμβο “SINK”, προσπελαύνει, μέσω μίας δομής επανάληψης, όλα τα boards, τα οποία είναι συνδεδεμένα, στέλνει αίτημα λήψης δεδομένων σε κάθε ένα και διαβάζει τα δεδομένα τα οποία λαμβάνει. Αυτή η διαδικασία επαναλαμβάνεται ανά ένα ορισμένο χρονικό διάστημα. Επίσης, δεδομένου ότι δεν υπάρχει διαφορετικό κανάλι εξόδου – εξαγωγής των δεδομένων σε εξωτερική εφαρμογή, τα δεδομένα τυπώνονται στο serial monitor του κόμβου.

```
#include <Wire.h>
...
void loop()
{
    ...
    Serial.print("fc23e,");
    for (int beehive = 1; beehive <= 4; beehive++)
    {
        digitalWrite(LED, HIGH); // Turn on the LED indicator

        Wire.requestFrom(beehive, 15); // Request 15 bytes from current beehive

        while (Wire.available())
        { // Slave may send less than requested
            char c = Wire.read(); // Receive a byte as character
            if (c != EOF)
                Serial.print(c);
        }
        Serial.print(',');
    }
    ...
    delay(SENDFREQ / 2);
}
```

Απόσπασμα_Κώδικα 2: Προσπέλαση των συνδεδεμένων boards, αποστολή αιτημάτων λήψης και λήψη των τιμών των αισθητήρων τους. - Αρχείο: “beehive_control_kit_sink.ino”.

Από το serial monitor του κόμβου “SINK”, μέσω της επέκτασης για chrome browsers “[Gext-tinkercad](#)”, λαμβάνονται, ανά ένα ορισμένο χρονικό διάστημα, τα περιεχόμενά του και στέλνονται σε έναν HTTP server, μέσω ενός POST αιτήματος.

```
var post_data = JSON.stringify({'source': window.location.href,
'value':latest_value});
$.ajax({
  type: "POST",
  contentType: 'application/json',
  url: post_endpoint,
  dataType: 'json',
  data: post_data
})
.done(function(){ log(myUUID, 'POST success.', post_data); })
.fail(function(xhr, status, error) { log(myUUID, 'POST fail: ', status, error,
xhr); });
```

Απόσπασμα_Κώδικα 3: Αποστολή των περιεχομένων του serial monitor σε έναν HTTP server μέσω POST request - Αρχείο: “contentScript.js”.

Στην πράξη, ο HTTP server, στον οποίο αποστέλλονται τα δεδομένα, είναι μία [Flask](#) web server εφαρμογή (Python Flask Application), η οποία εκτελείται σε ένα Amazon EC2 (Amazon Elastic Compute Cloud) instance, με λειτουργικό σύστημα “Ubuntu-18.04-amd64-server” και “ακούει” στην θύρα 1024. Ο εξυπηρετητής αυτός, μόλις λάβει ένα POST request, απομονώνει την τιμή “value” και τη στέλνει (publish) σε ένα topic, με όνομα τον κωδικό “kitId” του τρέχοντος Kit (εμπεριέχεται στα δεδομένα), σε έναν MQTT Broker.

```
@app.route('/post', methods=['GET', 'POST'])
def post():
    ...
    request_data = request.json

    # Get kit's id
    kId = request_data['value'].split(',')[0]

    # Publish recieved data to mqtt topic based on kit's id
    client.publish(str(kId), str(request_data['value']))

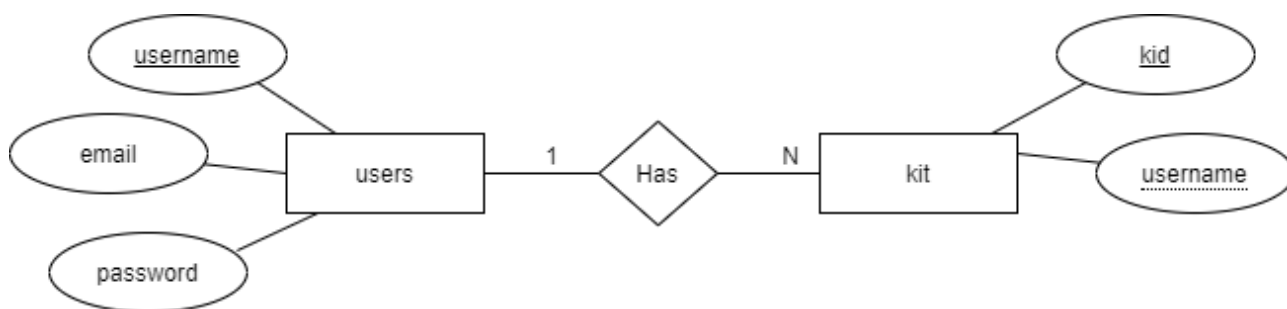
    return json.dumps(request_data)
```

Απόσπασμα_Κώδικα 4: Δημοσίευση των δεδομένων του POST request σε topic του Broker - Αρχείο: “main.py”.

Τα POST requests στον Flask Server γίνονται μέσω jQuery (βιβλιοθήκη της JavaScript). Το γεγονός αυτό ενεργοποιεί τον μηχανισμό ασφάλειας του ec2 instance, “Cross-Origin Resource Sharing” (CORS), με αποτέλεσμα να μπλοκάρει αυτά τα αιτήματα. Για να αποφευχθεί αυτό, χρειάζεται η δημιουργία ενός API, μέσω του Amazon API Gateway, το οποίο θα λειτουργεί ως μεσολαβητής για τα αιτήματα που θα γίνονται. Με αυτόν τον τρόπο μπορούν να οριστούν οι κατάλληλοι κανόνες ώστε να αποτρέπεται το μπλοκάρισμα των επιθυμητών αιτημάτων (αιτημάτων του browser extension).

Ο MQTT Broker, ένας Eclipse [Mosquitto](#) Message Broker, ο οποίο εκτελείται στο ίδιο ec2 instance και εκτελεί το πρωτόκολλο MQTT στην θύρα 1883. Επίσης, προκειμένου να μπορεί να επικοινωνεί η web εφαρμογή των χρηστών με τον Broker, δηλαδή να μπορεί να γίνεται λήψη των δεδομένων απευθείας από τον browser (μέσω JavaScript), ο Broker έχει ρυθμιστεί (επεμβαίνοντας στο αρχείο “/etc/mosquitto/conf.d/default.conf”) ώστε να εκτελεί το πρωτόκολλο επικοινωνίας WebSockets, μέσω της θύρας 9001.

Τέλος, σε ότι αφορά το κομμάτι της εφαρμογής του χρήστη, πρόκειται για μια web εφαρμογή γραμμένη σε PHP, σε συνδυασμό με μία σχεσιακή βάση δεδομένων MySQL. Ο εξυπηρετητής, ο οποίος φιλοξενεί την εφαρμογή είναι ένας [Apache HTTP](#) server, εγκατεστημένος, μαζί με τον MySQL Server, στο ίδιο ec2 instance και “ακούει” στην προεπιλεγμένη, για το πρωτόκολλο HTTP, θύρα 80. Μέσω της εφαρμογής, ο χρήστης μπορεί να δημιουργήσει έναν λογαριασμό, να συνδεθεί σε έναν ήδη υπάρχον, να προσθέσει τα “Beehive Control Arduino Kit” του, ώστε να αντιστοιχιστούν με τον λογαριασμό του και να επιλέγει από ποιο Kit επιθυμεί να παρακολουθεί κάθε φορά, τις τιμές των αισθητήρων του.




Εικόνα 6: Διάγραμμα Οντοτήτων-Συσχετίσεων (ER) της βάσης δεδομένων της εφαρμογής.

```

<form method="post" action="index.php">
  <div class="form-group">
    <label for="username">Username:</label>
    <input type="text" class="form-control" id="username" name="username" required>
  </div>
  ...
<?php
if(isset($_POST["username"], $_POST["password"])) {
  // Get the password for the given username
  $sql = "SELECT password FROM users WHERE username = '".$_POST["username"]."'";
  $hashed_pass = mysqli_query($conn, $sql)->fetch_assoc(); // Execute query
  if(!empty($hashed_pass["password"]))
    $hashed_pass = $hashed_pass["password"];

  // Check if given password (hash) value is equal to the password stored in db
  if (password_verify($_POST["password"], $hashed_pass)) {
    // Create a cookie with user's username
    setcookie("user", $_POST["username"], time() + (86400 * 30), "/");
    header("Location: home.php"); // Redirect to home.php
  } else
    echo "<br><br><p style='color: darkred'>Wrong Username or Password</p>";
}
?>
</form>
  
```

Απόσπασμα_Κώδικα 5: Sign in form - Ενδεικτικός κώδικας: ο τρόπος με τον οποίο λειτουργεί η web εφαρμογή – Αρχείο: “index.php”.




Username:

Password:

[Sign in](#)

Don't have an account? [Create one here!](#)

Εικόνα 7: Διεπαφή Web Εφαρμογής: Φόρμα σύνδεσης χρήστη.



Username:

Email:

Password:

Confirm Password:

☐ I agree to the [Terms & Conditions](#)

[Sign me up!](#)

Already have an account? [Sign in here!](#)

Εικόνα 8: Διεπαφή Web Εφαρμογής: Φόρμα δημιουργίας λογαριασμού.

Beehive Control theod_papadak

[← Back](#)

Add new Kit:
Insert the 5 letter code of your Kit, in order to link it with your account.

Kit ID:

[Add](#)

My Kits:
There was no Kit linked with your account.

A project of
Pashalis Grivas · George Gerachakis & George Melidoniotis

Beehive Control | Copyright ©2021

Εικόνα 9: Διεπαφή Web Εφαρμογής: Ο χρήστης προσθέτει στον λογαριασμό του τον αναγνωριστικό κωδικό (Kit ID) του Kit του.

Ο χρήστης, έχοντας προσθέσει τα Kit του στον λογαριασμό του, μέσω ενός dropdown menu στην αρχική οθόνη, μπορεί πλέον να τα επιλέξει ώστε να εμφανίζονται οι πληροφορίες των συγκεκριμένων κυψελών. Στο παρασκήνιο, αυτό το οποίο συμβαίνει είναι η εκτέλεση κώδικα JavaScript, ο οποίος χρησιμοποιώντας την βιβλιοθήκη [Eclipse Paho JavaScript Client](#), αρχικά εδραιώνει μία σύνδεση με τον MQTT Broker μέσω WebSockets, κάνει εγγραφή (subscribe) στο

συγκεκριμένο topic του επιλεγμένου Kit, και περιμένει να λάβει το επόμενο μήνυμα, το οποίο θα δημοσιευθεί στο topic. Με το που λαμβάνεται κάποιο μήνυμα (αλφαριθμητικό με δεδομένα των αισθητήρων), γίνεται απομόνωση/διαλογή των τιμών (parsing), γίνεται εμφάνιση των τιμών, παράγοντας επιπρόσθετο κώδικα HTML, και γίνεται έλεγχος των τιμών των αισθητήρων προκειμένου να διαπιστωθεί αν αυτές βρίσκονται εκτός των ορισμένων φυσιολογικών ορίων. Σε μία τέτοια περίπτωση δημιουργούνται και εμφανίζονται ειδοποιήσεις (alerts) στον χρήστη, με το κατάλληλο κάθε φορά μήνυμα.

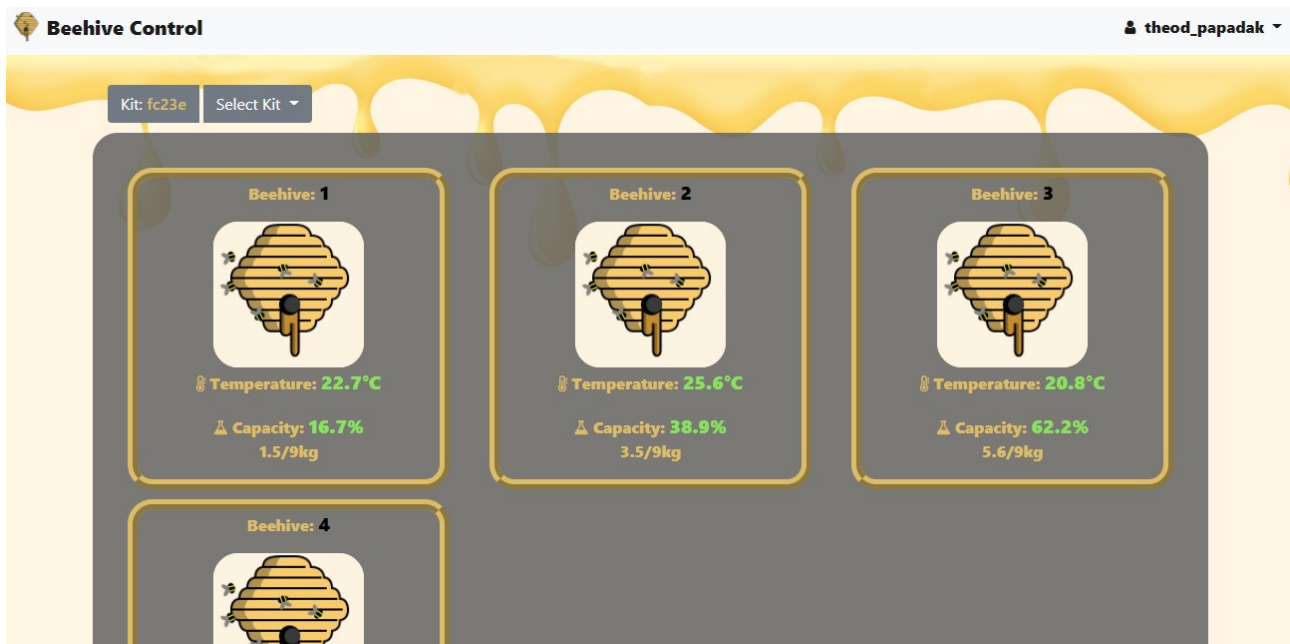
```
client.onMessageArrived = function (message) {
    data = message.payloadString.split(",");
    mainframe.innerHTML = "";
    for (let i=1; i<data.length-1; i++) {
        beehive = data[i].split(":")[0];
        temperature = data[i].split(":")[1];
        weight = data[i].split(":")[2];
        ...

        if (parseFloat(temperature) < 10 || parseFloat(temperature) > 35) {
            temperature_color = danger_color;
            alert("An extreme temperature of "+temperature+
                "°C was spotted in beehive "+beehive+"!");
        }
        else if (parseFloat(temperature) < 16 || parseFloat(temperature) > 27) {
            temperature_color = warning_color;
            alert("Temperature of "+temperature+
                "°C was spotted in beehive "+beehive+".");
        }

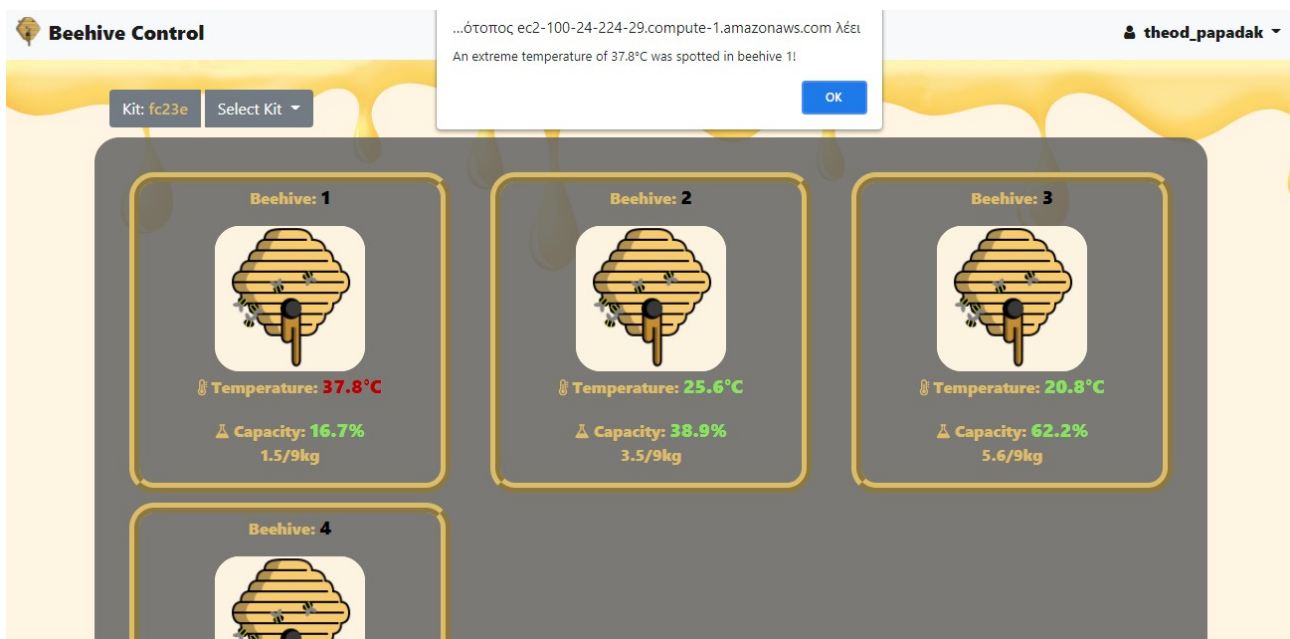
        if (parseFloat(weight) > 8) {
            weight_color = danger_color;
            alert("Beehive "+beehive+" is almost full! \n Capacity: "+
                Math.round(parseFloat(weight) / 9 * 100 * 10) / 10+"%");
        }
        else if (parseFloat(weight) > 6) {
            weight_color = warning_color;
            alert("Beehive "+beehive+" is about to get full! \n Capacity: "+
                Math.round(parseFloat(weight) / 9 * 100 * 10) / 10+"%");
        }

        mainframe.innerHTML += ...
        '<p>Beehive: <span style="color: black; font-size: larger">'
        +beehive+'</span></p>'+ ...
        +<span style="color: '+temperature_color+';font-size:larger">'
        +temperature+'°C</span></p>'+ ...;
    }
}
```

Απόσπασμα_Κώδικα 6: Λήψη των δεδομένων από το topic, parsing, εμφάνιση στη διεπαφή, έλεγχος και εμφάνιση προειδοποιητικών μηνυμάτων – Αρχείο: "home.php".



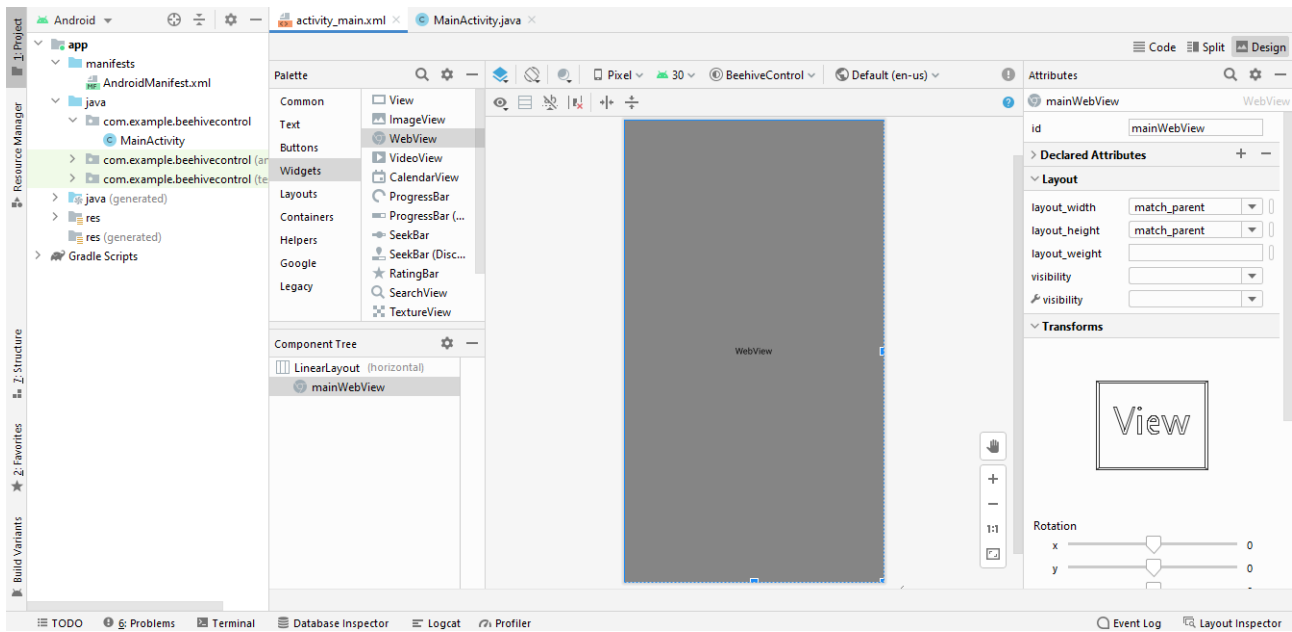
Εικόνα 10: Διεπαφή Web Εφαρμογής: Εμφάνιση των τιμών των αισθητήρων του επιλεγμένου Kit.



Εικόνα 11: Διεπαφή Web Εφαρμογής: Ειδοποίηση χρήστη για τιμές εκτός των φυσιολογικών ορίων.

Το γεγονός ότι έχει ακολουθηθεί μία mobile-first σχεδίαση (χρησιμοποιώντας το framework [Bootstrap 4](#)) της διεπαφής του χρήστη, δηλαδή τα στοιχεία της εφαρμογής ανταποκρίνονται και προσαρμόζονται κατάλληλα στο μέγεθος της οθόνης (responsive design), επιτρέπει την εύκολη μετατροπή της σε μία εφαρμογή για κινητές συσκευές Android. Προκειμένου να γίνει αυτό, χρησιμοποιώντας το περιβάλλον ανάπτυξης [Android Studio 4.2.1](#), προστέθηκε στη βασική οθόνη

της εφαρμογής (main activity) ένα “WebView” στοιχείο, το οποίο παρέχει τις βασικές λειτουργίες ενός browser. Χρησιμοποιώντας κώδικα Java, ορίζεται στο στοιχείο να ανοίγει την web εφαρμογή, μέσω της διεύθυνσής της, εκμεταλλεύοντας έτσι όλη την ήδη υλοποιημένη λειτουργικότητά της. Τέλος, δεδομένου ότι το “WebView” στοιχείο δεν υποστηρίζει τις ειδοποιήσεις (alerts) της JavaScript, προστέθηκε κώδικας για την ανίχνευση τους και την μετατροπή τους σε native διαλόγους ειδοποίησης (alert dialogs).

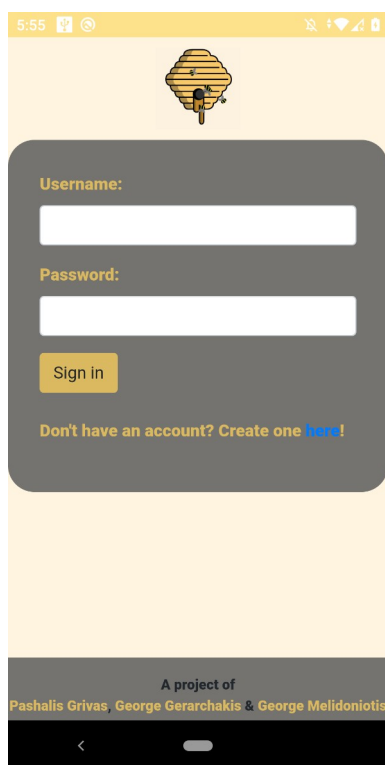


Εικόνα 12: Προσθήκη του WebView στοιχείου στην οθόνη (activity) της εφαρμογής.

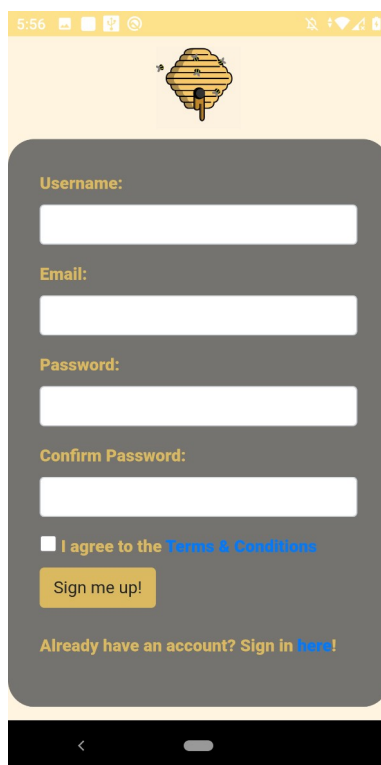
```
@Override
protected void onCreate(Bundle savedInstanceState) {
    ...
    WebView browser = (WebView) findViewById(R.id.mainWebView);
    browser.setWebViewClient(new WebViewClient()); // Open urls inside browser
    browser.getSettings().setDomStorageEnabled(true); // Enable local storage
    browser.setVerticalScrollBarEnabled(true); // Enable vertical scrolling
    browser.getSettings().setJavaScriptEnabled(true); // Enable javascript

    // Turn JS alerts into alertboxes
    browser.setWebChromeClient(new WebChromeClient() {
        @Override
        public boolean onJsAlert(WebView view, String url, String message,
                                JsResult result) {
            new AlertDialog.Builder(myApp)
                .setTitle("Beehive Alert")
                .setMessage(message)
                .setPositiveButton(android.R.string.ok, (dialog, which) ->
                                result.confirm())
                .setCancelable(false)
                .create()
                .show();
            return true;
        }
    });
    browser.loadUrl("http://ec2-100-24-224-29.compute-1.amazonaws.com/bee hive");
}
```

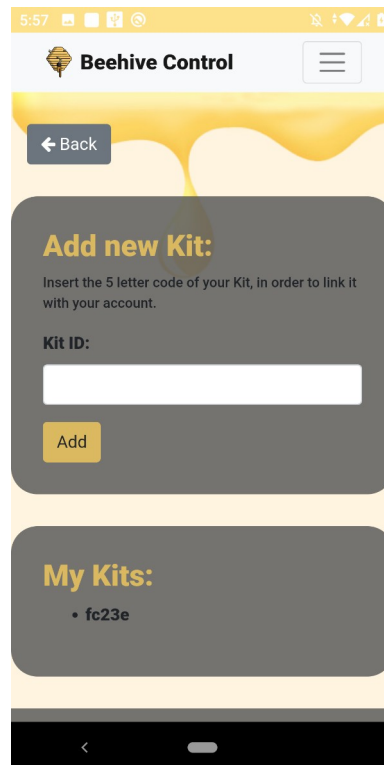
Απόσπασμα_Κώδικα 7: Ορισμός της διεύθυνσης της web εφαρμογής και μετατροπή των Js alerts σε Android AlertDialogs – Αρχείο: “MainActivity.java”.



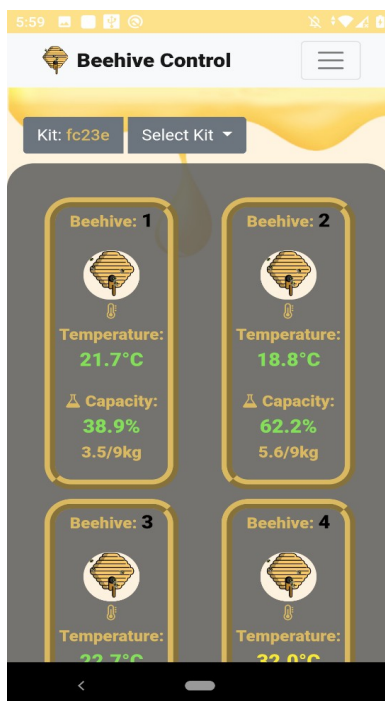
Εικόνα 13: Διεπαφή Android Εφαρμογής: Sing-in.



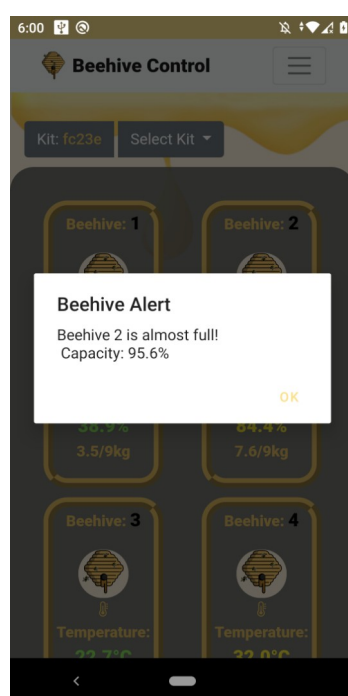
Εικόνα 14: Διεπαφή Android Εφαρμογής: Sign-up.



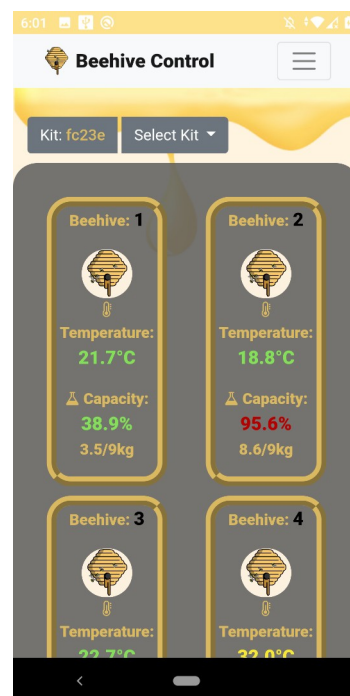
Εικόνα 15: Διεπαφή Android Εφαρμογής: Προσθήκη Kit.



Εικόνα 16: Διεπαφή Android Εφαρμογής: Τιμές αισθητήρων του επιλεγμένου Kit.



Εικόνα 17: Διεπαφή Android Εφαρμογής: Ειδοποίηση χρήστη μέσω "alertDialog".



Εικόνα 18: Διεπαφή Android Εφαρμογής: Τιμές αισθητήρων του επιλεγμένου Kit – Ενδειξη για ακραίες τιμές.

Κεφάλαιο 3

Αξιολόγηση - Συμπεράσματα

3.1 Τρόπος Αξιολόγησης της Εφαρμογής

Προκειμένου να γίνει αξιολόγηση της εφαρμογής, σε πρώτο επίπεδο μπορεί να διεξαχθεί μία έρευνα, μέσω ερωτηματολογίων ή συνεντεύξεων. Με αυτόν τον τρόπο, μπορεί να αποτυπωθεί η γνώμη των επαγγελματιών μελισσοκόμων, σχετικά με το αν μία τέτοια εφαρμογή μπορεί να επιτύχει τον σκοπό της και να συνεισφέρει αποτελεσματικά. Επίσης, μία άλλη προσέγγιση για την αξιολόγηση της προτεινόμενης λύσης, είναι η δοκιμαστική εφαρμογή της. Εγκαθιστώντας το “Beehive Control Arduino Kit” σε πραγματικές κυψέλες, θα διαπιστωθούν τυχόν προβλήματα, τα οποία μπορεί να υπάρχουν κάτω από ρεαλιστικές συνθήκες, ενώ η απόδοση κατά την δοκιμαστική περίοδο μπορεί να συγκριθεί με την απόδοση μίας προηγούμενης περιόδου, ώστε η αποτελεσματικότητα της εφαρμογής να αποτυπωθεί σε πραγματικά νούμερα.

3.2 Τεχνολογικοί και Άλλοι Περιορισμοί

Ένα σημαντικό πρόβλημα, το οποίο εμπεριέχει η προτεινόμενη λύση, είναι το γεγονός ότι η εγκατάσταση ενός τέτοιου Kit, ακόμα και με την παροχή αναλυτικών οδηγιών, απαιτεί μία σχετική τεχνολογική εξοικείωση και επιδεξιότητα από μεριάς του επαγγελματία μελισσοκόμου. Αυτό, σε πραγματικές συνθήκες, είναι ένας αποθαρρυντικός - περιοριστικός παράγοντας για πολλούς επαγγελματίες.

Σημαντικό περιορισμό, αποτελεί επίσης το γεγονός ότι όσο αυξάνεται ο αριθμός των κυψελών, τόσο αυξάνεται και το κόστος αγοράς του υλικού εξοπλισμού της εφαρμογής. Δεδομένου ότι ο κόμβος “SINK” μπορεί να υποστηρίξει έναν συγκεκριμένο αριθμό συνδεδεμένων συσκευών, απαιτείται η αγορά επιπρόσθετων τέτοιων κόμβων, πέρα από των μικροελεγκτών και εξαρτημάτων της κάθε κυψέλης.

3.3 Συμπεράσματα και Προοπτικές Μελλοντικής Αξιοποίησης

Η εφαρμογή “Beehive Control”, μπορεί να αποτελέσει μία ουσιαστική λύση στα προβλήματα της διαδικασίας ελέγχου και παρακολούθησης των κυψελών, τα οποία αντιμετωπίζει συχνά ένας επαγγελματίας μελισσοκόμος. Τα οφέλη από την εφαρμογή αφορούν τόσο την απόδοση των μελισσών, την αύξηση της παραγωγικότητας και την μείωση του κόστους παραγωγής, όσο και την βελτίωση της προσωπικής ζωής του μελισσοκόμου, καθώς επιτυγχάνεται σημαντική εξοικονόμηση χρόνου. Από την άλλη, προκύπτουν κάποια πρακτικά προβλήματα, τα οποία λειτουργούν ως αποτρεπτικοί παράγοντες, για πολλούς επαγγελματίες μελισσοκόμους, για την εφαρμογή μίας τέτοιας λύσης, με το κυριότερο να είναι οι σχετικές τεχνικές γνώσεις, οι οποίες απαιτούνται.

Τα περιθώρια βελτίωσης της προτεινόμενης λύσης είναι αρκετά, καθώς από την μία μπορεί να γίνει ένας καλύτερος σχεδιασμός της ενώ από την άλλη μπορεί να ολοκληρωθεί και να βελτιωθεί η υλοποίησή των εφαρμογών της. Πιο συγκεκριμένα, δεδομένου ότι οι web και android εφαρμογές βρίσκονται σε πρώιμο στάδιο ανάπτυξης (demos), πολλές από τις λειτουργίες, οι οποίες περιγράφηκαν στις ενότητες των προηγούμενων κεφαλαίων, δεν έχουν υλοποιηθεί. Για παράδειγμα, αποθηκεύοντας τις τιμές στη βάση δεδομένων, μπορεί να δίνεται η δυνατότητα να προβάλλονται διάφορα διαγράμματα, να υπολογίζονται σύνθετοι δείκτες και να γίνεται μία καλύτερη διαχείριση των κυψελών και της παραγωγής.

Αναφορές – Πηγές

- Μελισσοκόμος, Συμβουλευτική – Επαγγελματικός Προσανατολισμός: <http://edujob.gr/node/274>
- Arduino Scale With 5kg Load Cell and HX711 Amplifier:
<https://www.instructables.com/Arduino-Scale-With-5kg-Load-Cell-and-HX711-Amplifi/>
- Add WiFi to Arduino UNO, Arduino Project Hub:
<https://create.arduino.cc/projecthub/imjeffparedes/>
- Solar Charged Battery Powered Arduino Uno, Arduino Project Hub:
<https://create.arduino.cc/projecthub/>
- Eclipse Paho JavaScript Client:
<https://www.eclipse.org/paho/index.php?page=clients/js/index.php>
- MQTT over websocket in python, Stack Overflow: <https://stackoverflow.com/a/37678415>
- Enabling CORS for a REST API resource:
<https://docs.aws.amazon.com/apigateway/latest/developerguide/how-to-cors.html>
- Gext-tinkercad: <https://github.com/riggas-ionio/smart-iot/tree/master/Gext-tinkercad>
- JavaScript alert not working in Android WebView, Stack Overflow:
<https://stackoverflow.com/questions/5271898/javascript-alert-not-working-in-android-webview>