ΠΑΝΕΠΙΣΤΗΜΙΟ ΙΩΑΝΝΙΝΩΝ

ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ
ΜΗΧΑΝΙΚΩΝ Η/Υ & ΔΙΚΤΥΩΝ

ΠΑΝΕΠΙΣΤΗΜΙΟ ΙΩΑΝΝΙΝΩΝ

ΣΧΟΛΗ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ ΤΜΗΜΑ

ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΠΜΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΔΙΚΤΥΩΝ

ΜΕΤΑΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΠΛΑΤΦΟΡΜΑ ΥΠΟΛΟΓΙΣΤΙΚΟΥ ΝΕΦΟΥΣ ΓΙΑ ΤΗΝ ΕΠΕΞΕΡΓΑΣΙΑ ΙΣΤΟΠΑΘΟΛΟΓΙΚΩΝ ΕΙΚΟΝΩΝ

Οντισέ Τσακάι

Επιβλέπων: Νικόλαος Γιαννακέας,

Επίκουρος Καθηγητής

Άρτα, Σεπτέμβριος, 2020

ΠΑΝΕΠΙΣΤΗΜΙΟ ΙΩΑΝΝΙΝΩΝ

ΠΑΝΕΠΙΣΤΗΜΙΟ ΙΩΑΝΝΙΝΩΝ

ΣΧΟΛΗ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ ΤΜΗΜΑ

ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΠΜΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΔΙΚΤΥΩΝ

ΜΕΤΑΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΠΛΑΤΦΟΡΜΑ ΥΠΟΛΟΓΙΣΤΙΚΟΥ ΝΕΦΟΥΣ ΓΙΑ ΤΗΝ ΕΠΕΞΕΡΓΑΣΙΑ ΙΣΤΟΠΑΘΟΛΟΓΙΚΩΝ ΕΙΚΟΝΩΝ

Οντισέ Τσακάι

Επιβλέπων: Νικόλαος Γιαννακέας,

Επίκουρος Καθηγητής

Άρτα, Σεπτέμβριος, 2020

CLOUD-BASED PLATFORM FOR PROCESSING OF HISTOPATHOLOGICAL IMAGES

Εγκρίθηκε από τριμελή εξεταστική επιτροπή

Άρτα, Σεπτέμβριος 2020

ΕΠΙΤΡΟΠΗ ΑΞΙΟΛΟΓΗΣΗΣ

1. Επιβλέπων καθηγητής

   Νικόλαος Γιαννακέας,

   Επίκουρος Καθηγητής

2. Μέλος επιτροπής

   Χρήστος Γκόγκος,

   Αναπληρωτής Καθηγητής

3. Μέλος επιτροπής

   Αλέξανδρος Τζάλλας,

   Επίκουρος Καθηγητής

Ο Διευθυντής του ΠΜΣ

Χρυσόστομος Στύλιος,

Καθηγητής Α’ Βαθμίδας,

Υπογραφή

Δήλωση μη λογοκλοπής

Δηλώνω υπεύθυνα και γνωρίζοντας τις κυρώσεις του Ν. 2121/1993 περί Πνευματικής Ιδιοκτησίας, ότι η παρούσα μεταπτυχιακή εργασία είναι εξ ολοκλήρου αποτέλεσμα δικής μου ερευνητικής εργασίας, δεν αποτελεί προϊόν αντιγραφής ούτε προέρχεται από ανάθεση σε τρίτους. Όλες οι πηγές που χρησιμοποιήθηκαν (κάθε είδους, μορφής και προέλευσης) για τη συγγραφή της περιλαμβάνονται στη βιβλιογραφία.

Τσακάι, Οντισέ

Υπογραφή

ΕΥΧΑΡΙΣΤΙΕΣ

Προτού ξεκινήσει η ανάπτυξη του θέματος της παρούσας μεταπτυχιακής εργασίας, θεωρώ χρέος μου να ευχαριστήσω όλους όσοι συνέβαλαν σημαντικά στην εκπόνησή της, με την διαρκή υποστήριξή τους. Πρωτίστως επιθυμώ να ευχαριστήσω τον επιβλέποντα καθηγητή της διπλωματικής εργασίας, κ. Νικόλαο Γιαννακέα, για την ανάθεση του θέματος, την πολύτιμη καθοδήγησή του και την εμπιστοσύνη που έδειξε στο πρόσωπό μου. Στη συνέχεια, θα ήθελα να ευχαριστήσω τους κ. Αλέξανδρο Τζάλλα, κ. Χρήστο Γκόγκο, μέλη της τριμελούς επιτροπής καθώς και τον κ. Μάρκο Τσίπουρα, Αναπληρωτής Καθηγητή του τμήματος Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών του Πανεπιστημίου Δυτικής Μακεδονίας, για την καθοριστική συμβολή τους στην εκπόνηση της μεταπτυχιακής μου εργασίας.

ΠΕΡΙΛΗΨΗ

Στο πλαίσιο της παρούσας εργασίας αναπτύχθηκε μια καινοτόμος πλατφόρμα διαδικτυακού νέφους για την κατάτμηση εικόνων. Η ιδέα ξεκίνησε από την ενασχόληση της ερευνητικής ομάδας με την επεξεργασία ιστοπαθολογικών εικόνων βιοψιών ήπατος, με σκοπό την εύρεση παθολογικών ευρημάτων και την ποσοτικοποίηση τους ως ποσοστό του φυσιολογικού ιστού. Ωστόσο, κατά την ανάπτυξη κατέστη εμφανές το γεγονός ότι η υποδομή που δημιουργήθηκε, δύναται να αξιοποιηθεί σε ποικίλα προβλήματα κατάτμησης, αρκεί ο εξειδικευμένος χρήστης κάθε φορά να παρέχει την γνώση στην πλατφόρμα, επισημειώνοντας τα αντικείμενα ενδιαφέροντος. Τα πλεονεκτήματα της πλατφόρμα συνοψίζονται στα εξής:

- Έχει υλοποιηθεί με τεχνολογίες ανάπτυξης στο υπολογιστικό νέφος και επομένως αξιοποιεί τους πόρους του. Πρακτικά, δίνει την δυνατότητα στον χρήστη να εργάζεται στην πλατφόρμα κάνοντας χρήση μόνο του φυλλομετρητή του.
- Έχει δυνατότητα φιλοξενίας πολλαπλών πιστοποιημένων χρηστών
- Περιέχει εύχρηστο επισημειωτή με σκοπό να καταχωρούνται αντικείμενα ενδιαφέροντος για την εκπαίδευση συστήματος Τεχνητής Νοημοσύνης
- Το σύστημα τεχνητής νοημοσύνης είναι βασισμένο στην Βαθιά Μάθηση, και συγκεκριμένα στην τεχνολογία των Συνεκτικών Νευρωνικών Δικτύων με μάσκα.
- Δέχεται πολλαπλές εκπαιδεύσεις από τους διαφορετικούς χρήστες αλλά μπορεί να ενημερώνει και να ενισχύει τις εκπαιδεύσεις με βάση νέα επισημειωμένα ευρήματα.
- Η εκπαίδευση των Συνεκτικών Νευρωνικών Δικτύων αξιοποιεί την υπολογιστική ισχύ από κάρτες γραφικών με τεχνολογία CUDA.
- Είναι επεκτάσιμη σε πληθώρα ερευνητικών πεδίων κατάτμησης εικόνων, πέραν της κατάτμησης βιοψιών ήπατος, όπου δοκιμάστηκε πιλοτικά στο πλαίσιο της μεταπτυχιακής εργασίας

Το κείμενο της εργασίας επιχειρεί να αποσαφηνίσει της τεχνικές λεπτομέρειες της υλοποίησης της πλατφόρμας, από την διαδικασία της καταγραφής των απαιτήσεων, την αρχιτεκτονική του συστήματος και τις τεχνολογίες που χρησιμοποιήθηκαν, έως και τις διεπαφές με τον χρήστη.

Λέξεις-κλειδιά: Πλατφόρμα νέφους, Υπολογιστικό νέφος, Συνελικτικά Νευρωνικά Δίκτυα, ιστοπαθολογικές εικόνες

ABSTRACT

In this work, an innovative cloud platform for image segmentation has been developed. According to the concept of the study image processing of histopathological images is utilized, to detect pathological findings as a percentage of normal tissue in liver biopsies. The developed infrastructure can be employed in a variety of different segmentation problems, as long as the specialized user provides knowledge to the platform, annotating the objects of interest. The advantages of the platform could be summarized as follows:

- It has been implemented based on cloud computing technologies and therefore utilizes cloud resources. As a result, the user is able to work on the platform using only his browser
- It is a multiuser platform where different users can be authenticated.
- It contains a user-friendly annotator, for finding input to feed the Artificial Intelligence system
- The Artificial intelligent system is based on Deep Learning and more specifically on Convolutional Neural Networks CNN
- The training of CNN utilizes resources from Graphic Processor Unit (GPU) using CUDA technology.
- It is expandable in many segmentation problems, apart from the histopathological image segmentation.

The manuscript attempts to describe and clarify technical details of the implementation of the platform, starting with the process of user requirements, the system architecture and the technologies used, and ending with the description of user interfaces.

Keywords: Cloud-based Platform, Convolutional Neural Networks, Histopathological images.

Περιεχόμενα

ΚΑΤΑΛΟΓΟΣ ΔΙΑΓΡΑΜΜΑΤΩΝ/ΕΙΚΟΝΩΝ

ΚΑΤΑΛΟΓΟΣ ΔΙΑΓΡΑΜΜΑΤΩΝ/ΕΙΚΟΝΩΝ

ΠΙΝΑΚΑΣ ΣΥΝΤΟΜΟΓΡΑΦΙΩΝ

MVC……. Model-View-Controller

ΤΕΙ-Η..……..Τεχνολογικό Εκπαιδευτικό Ίδρυμα Ηπείρου

Κεφάλαιο 1

1. Τεχνολογικό Υπόβαθρο

Η δημιουργία μια λειτουργικής πλατφόρμας στο διαδικτυακό νέφος αναμφισβήτητα αποτελεί μια πολύπλοκη διαδικασία, από τον σχεδιασμό έως την υλοποίηση της, στην οποία εμπλέκονται πολλές διαφορετικές τεχνολογίες. Ενδεικτικά, υπάρχουν απαιτήσεις για την δημιουργία του συστήματος υποστήριξης (backend), της βάσης δεδομένων, των διεπαφών των χρηστών κ.ο.κ.. Στο παρόν κεφάλαιο εξετάζονται και αναλύονται οι τεχνολογίες που χρησιμοποιήθηκαν για την δημιουργίας της πλατφόρμας και οι οποίες είναι απαραίτητες για τον αναγνώστη να γνωρίζει ως τεχνολογικό υπόβαθρο. Ως εκ τούτου ο αναγνώστης θα κατανοήσει την προσέγγιση της σχεδίασης της αρχιτεκτονικής καθώς και της λεπτομέρειες της εκτέλεσης της εφαρμογής.

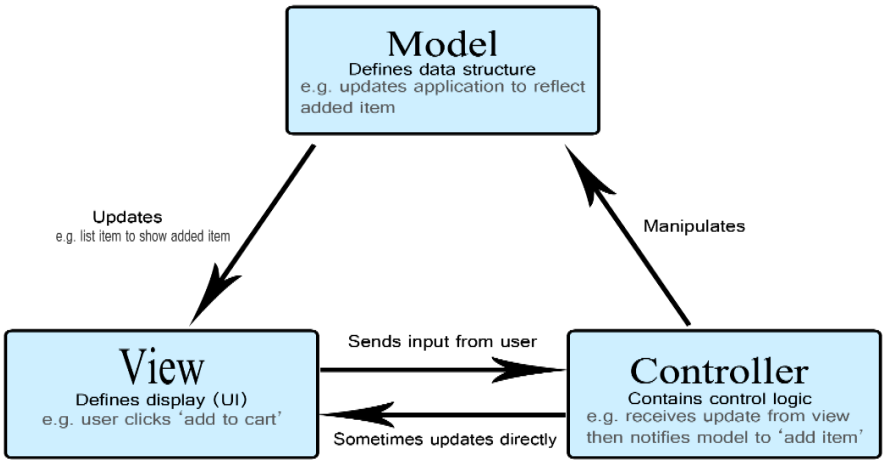
1.1 Αρχιτεκτονική Μοντέλου-Όψης-Ελεγκτή

Η αρχιτεκτονική Μοντέλου-Όψης-Ελεγκτή (Model-View-Controller - MVC) είναι μια γνωστή διαδεδομένη αρχιτεκτονική δόμησης του πηγαίου κώδικα, η οποία παρουσιάστηκε στην αντικειμενοστραφής γλώσσα προγραμματισμού Smalltalk’80 από τον Krasner και τον Ripe [1]. Πιο συγκεκριμένα, δημοσιεύτηκε σε επιστημονικό άρθρο το 1979 από έναν από τους δημιουργούς της γλώσσας Smalltalk, τον Trygve ReenSkauge. Η αρχιτεκτονική αποτελεί ένα βασικό συστατικό της εφαρμογής καθώς αξιοποιεί το πλαίσιο λογισμικού (framework) Yii2 αλλά και γνωστά και ευρέως χρησιμοποιούμενα πλαίσια. Η αρχιτεκτονική αποτελείται από τους τρεις βασικούς πυλώνες, α) το Μοντέλο (Model), την όψη ή Γραφική Διεπαφή (View) και τον Ελεγκτή (Controller). Η παρακάτω συνιστώσες της αρχιτεκτονικής έχουν ως εξής:

- Κλάση Μοντέλου (Model Class): είναι μια αναπαράσταση των δεδομένων που δημοσιεύονται στον ελεγκτή, επίσης είναι η αντικειμενοστραφής αναπαράσταση μιας

οντότητα (ενός πίνακα) από την Βάση Δεδομένων. Το μοντέλο δεν εξαρτάται από τον ελεγκτή ή από όψη, αλλά είναι ανεξάρτητο.

- Συνιστώσα όψης (View Component): είναι υπεύθυνη για την λογική αναπαράστασης της γραφικής διεπαφή στον χρήστη. Εκτός από φόρτωση και απεικόνιση αρχείων, προετοιμάζει τα δεδομένα που λήφθηκαν από το μοντέλο, με σκοπό να μπορέσει να τα εμφανίσει. Το αποτέλεσμα της προετοιμασίας αυτής, ιδανικά θα πρέπει αν παρέχει πληροφορίες για τις μεταβλητές (Integer, Boolean κτλ.), για μπορούν ευκολά να μεταφερθούν στα αρχεία στο στάδιο της φόρτωσης και απεικόνισης.
- Κλάση ελεγκτή (Controller Class): είναι υπεύθυνη για να ελέγχει την ροή της εφαρμογής και δρα ως συντονιστής αναμεσά στο μοντέλο και την όψη. Όταν δημιουργείται κάποιο αίτημα (Request) σε μια εφαρμογή με βάση την αρχιτεκτονική MVC, θα πρέπει οπωσδήποτε να επιστραφεί μια απάντηση. Ωστόσο, υπάρχει η δυνατότητα και να εκτελέσει μια ή παραπάνω ενέργειές ώστε όταν δέχεται ένα συγκεκριμένο αίτημα να μπορεί να επιστρέφει διαφορετική απάντηση για το ίδιο αίτημα.



Εικόνα 1: Αρχιτεκτονική MVC

Η αρχιτεκτονική MVC αναπτύσσεται σε τρία επίπεδο και διαχωρίσει τα χαρακτηριστικά της εφαρμογής. Το πρώτο επίπεδο σχετίζεται με την λογική εισαγωγής του χρήστη, το δεύτερο επίπεδο αναφέρεται στην επιχειρησιακή λογική και το τρίτο επίπεδο χρησιμοποιείται για την διεπαφή του χρήστη. Επίσης η αρχιτεκτονική [2] παρέχει πολύ χαλαρή σύνδεση μεταξύ των τριών στρωμάτων, αλλά και την δυνατότητα παράλληλης ανάπτυξης στην ιδιά εφαρμογή. Αυτό σημαίνει ότι τα επίπεδα της εφαρμογής είναι

ανεξάρτητα μεταξύ τους, και επομένως τρεις προγραμματιστές μπορούν να εργάζονται ταυτόχρονα σε κάθε ένα από αυτά. Ένα προγραμματιστής μπορεί να δουλεύει στην λογική εισόδου χρήστη (Controller Class), άλλος να εργάζεται στην διεπαφή του χρήστη (View) και ένα τρίτος μπορεί να δουλεύει στην επιχειρησιακή λογική (Model Class) ταυτόχρονα.

Ενδεικτικά στην εφαρμογή η οποία αναπτύχθηκε στο πλαίσιο της εργασίας, στο σύστημα υποστήριξης (Backend) ένα μοντέλο αποτελεί το σχήμα (Shape) του ευρήματος που θα επισημειώσει ο ειδικός ιατρός, και το οποίο προέρχεται από μεταγλώττισή μέσω της σχεσιακή βάση δεδομένων σε αντικειμενοστραφή. Το σχήμα είναι από το πιο βασικά μοντέλα γιατί είναι υπεύθυνο για την εκχωρήσει και εμφανίσει όλων των μασκών σε όλες τις εικόνες της εφαρμογής. Όπως φαίνεται και στην παρακάτω εικόνα, υπάρχει η αντικειμενοστραφής συσχέτιση στην συνάρτηση GetImageShapes που συνδέει το αντικείμενο αυτό με το μοντέλο ImageShape. Το ImageShape συνδέει τον κάθε Shape με την συγκεκριμένη εικόνα της βάσης δεδομένων. Η σχεσιακή δομή των δυο μοντέλων είναι παραγόμενες από την βάση δεδομένων και η σύνδεση τους είναι «πολλά προς πολλά». Στην ουσία είναι μια συσχέτιση μεταξύ των τριών μοντέλων Εικόνα (Image), σχήμα (Shape) και σχήμα εικόνας (ImageShape).

```php
class Shape extends Model
{
    public static function tableName()
    {
        return '{{%shape}}';
    }

    public $temp_value;
    public $image_id;
    public $shape_deleted;
    public $shape_creator;
    public $testing_id;

    public function rules()
    {
        return [
            [['points', 'shape_type_id', 'class_id', "area"], 'required'],
            [['points'], 'string'],
            [['area'], 'number'],
            [['shape_type_id', 'class_id'], 'integer'],
            [['date_created', 'date_updated', 'date_born'], 'safe'],
            [['shape_type_id'], 'exist', 'skipOnError' => true, 'targetClass'
=> ShapeType::className(), 'targetAttribute' => ['shape_type_id' => 'id']],
            [['class_id'], 'exist', 'skipOnError' => true, 'targetClass' =>
Classification::className(), 'targetAttribute' => ['class id' => 'id']],
        ];
    }

    public function getImageShapes()
    {
        return $this->hasMany(ImageShape::className(), ['shape_id' => 'id']);
    }

    public function attributeLabels()
    {
        return [
            'id' => Yii::t('app', 'ID'),
            'points' => Yii::t('app', 'Points'),
            'area' => Yii::t('app', 'Area'),
            'shape_type_id' => Yii::t('app', 'Shape Type'),
            'testing_id' => Yii::t('app', 'Testing'),
            'class_id' => Yii::t('app', 'Classification'),
            'image_id' => Yii::t('app', 'Image'),
            'shape_deleted' => Yii::t('app', 'Deleted'),
            'shape_creator' => Yii::t('app', 'Creator'),
            'image_deleted' => Yii::t('app', 'Deleted'),
            'date_created' => Yii::t('app', 'Date Created'),
            'date_updated' => Yii::t('app', 'Date Updated'),
        ];
    }

    /**
     * @return \yii\db\ActiveQuery
     */

    public function getImages()
    {
```

Εικόνα 2:Μόντελο Shape και η συνάρτηση GetImageShapes.

2.1.1 Πλεονεκτήματα Αρχιτεκτονικής MVC

Τα βασικά πλεονεκτήματα της αρχιτεκτονικής συνοψίζονται στα εξής:

- **Διαχωρισμός πολυπλοκότητας:** Ο διαχωρισμός πολυπλοκότητας της εφαρμογής αποτελεί ένα από τα πιο βασικά πλεονεκτήματα της αρχιτεκτονικής MVC, καθώς διαχωρίσει την εφαρμογή, όπως προαναφέρθηκε, σε τρία διαφορετικά επίπεδα, τα

επίπεδα Model, View και Controller. Το κάθε επίπεδο εκτελεί διαφορικό έργο αλλά συνεργάζεται με τα αλλά δύο. Ως εκ τούτου, η δομή αυτή διευκολύνει την ταυτόχρονη ανάπτυξη του κώδικα από διαφορετικούς προγραμματιστές.

- **Ανάπτυξη οδηγούμενη από τον έλεγχο (Test-Driven Development):** Λογού της δομής της αρχιτεκτονικής, η ανάπτυξη οδηγούμενή από έλεγχο καθίσταται πιο εύκολη. Στην ουσία είναι μια διαδικασία ανάπτυξης λογισμικού που βασίζεται στην επανάληψη ενός πολύ συντόμου κύκλου υλοποίησης και οι απαιτήσεις μετατρέπονται σε πολύ συγκεκριμένες περιπτώσεις δοκίμων. Έπειτα ο κώδικας της εφαρμογής βελτιώνεται ώστε να με σκοπό να εξυπηρετούν τις δοκιμές αυτές.
- **Επεκτασιμότητα:** Είναι ένα από τα βασικά πλεονέκτημα της αρχιτεκτονικής. Η δομής της επιτρέπει την επεκτασιμότητα, δηλαδή την δυνατότητα να προσθέσουν παραπάνω λειτουργίες μελλοντικά. Ωστόσο, βοηθάει και στην τροποποίηση τρεχόντων λειτουργιών.
- **Πλήρης Έλεγχο:** Η αρχιτεκτονική MVC δεν χρησιμοποιεί φόρμες που βασίζονται στο διακομιστή (Server-Based Forms), για αυτό είναι ιδανική για τους προγραμματιστές που επιθυμούν να έχουν πλήρες έλεγχο στην συμπεριφορά της εφαρμογής.

Κατά αντιστοιχία η αρχιτεκτονική παρουσιάζει ορισμένα μειονεκτήματα:

- **Μεγάλη πολυπλοκότητα:** Κυρίως λόγω της δομής των επίπεδων του προτύπου MVC, η πολυπλοκότητα της εφαρμογής παρουσιάζεται μεγάλη. Επίσης αυξάνει την φύση των συμβάντων στον κώδικα της διεπαφής χρήστη, με αποτέλεσμα να καθιστά δύσκολη την αποσφαλμάτωση.
- **Μείωση απόδοσης μικρών εφαρμογών:** Το κύριο μειονεκτήματα της αρχιτεκτονικής MVC είναι το γεγονός ότι δεν είναι κατάλληλη για μικρές εφαρμογές. Λόγο της πολυπλοκότητας και του όγκου του προτύπου, μειώνει την απόδοση της εφαρμογής αλλά και των προγραμματιστών.

1.2 Το πλαίσιο λογισμικού Yii 2.0 της PHP

Το πλαίσιο λογισμικού (Framework) Yii [3] αποτελεί ένα μοντέρνο πλαίσιο ανοιχτού κώδικα

υλοποιημένο στη προγραμματιστική γλώσσα PHP, και παρέχεται ως πακέτο (Composer). Χρησιμοποιείται για την κατασκευή διαδικτυακών εφαρμογών και μεγάλου μεγέθους πλατφόρμων. Η αρχιτεκτονική του βασίζεται στην αρχιτεκτονική MVC η αναφέρθηκε και εξετάστηκε παραπάνω και είναι ένα σημαντικό κομμάτι της εφαρμογής που υλοποιήθηκε, καθώς με την τεχνολογία αυτή σχεδιαστικέ και υλοποιήθηκε το σύστημα υποστήριξης (Backend) της εφαρμογής. Στο κεφάλαιο αυτό θα εξεταστεί το πλαίσιο αναλυτικά, ώστε να γίνουν κατανοητές οι βασικές δυνατότητες και λειτουργικότητες του.

Εικόνα 3: Λογότυπο του Yii (https://www.yiiframework.com/)

Το πλαίσιο δημιουργήθηκε σε μια προσπάθεια για να καλύψει τα αρνητικά και ελαττώματα ενός προηγούμενου πλαισίου (PRADO Framework), το οποία και αυτό είναι ένα πλαίσιο ανοιχτού λογισμικού PHP Framework οδηγούμενο από συμβάντα. Η πρώτη εκδοχή του Yii (alpha version) δημιουργήθηκε το 2006, ενώ η υλοποίηση της διήρκησε 8 μήνες. Έπειτα από δυο χρονιά εκδόθηκε η επίσημη έκδοση 1.0. Το λογισμικού αυτό είναι ιδιαίτερα εύχρηστο και ευρέως χρησιμοποιούμενο και δημοφιλή για την ανάπτυξη διδακτικών εφαρμογών. Θεωρείται ένα ολοκληρωμένο λογισμικού και διαθέτει ένα εξαιρετικό εγχειρίδιο χρήσης, γεγονός που βοηθάει όλων των επίπεδων τους προγραμματιστές, από τον πιο αρχάριο έως τον πιο έμπειρο. Σκοπός και στόχος των προγραμματιστών ήταν η οργάνωση, η απλότητα και η σωστή διαχείριση των επιμέρους τμημάτων της εφαρμογής, με αποτέλεσμα να την υλοποίηση εύκολη και κατανοητή προς τον προγραμματιστή και αναγνώστη.

2.2.1 Δυνατότητες του Yii

Όσον αφορά τις δυνατότητες του, το πλαίσιο Yii διευκολύνει την ανάπτυξη των διαδικτυακών εφαρμογών μεγάλης πολυπλοκότητα, καθώς απλοποιεί τις διεργασίες που γίνονται επανειλημμένα και με είναι δύσκολες. Για παράδειγμα διευκολύνει ουσιαστικά και καθιστά απλές κάποιες σημαντικές εργασίες διαχείρισης της βάσης δεδομένων. Κάποιες βασικές δυνατότητες του πλαισίου περιγράφονται παρακάτω:

- Αξιοποιεί το πρότυπο MVC που αναφέρθηκε παραπάνω
- Δημιουργεί συνθέτες προδιαγραφές υπηρεσιών, ως είναι ένα XML πρότυπο (Web

Services Description Language – WSDL [4]) που χρησιμοποιείται για να περιγράψει την λειτουργικότητα και να διαχειριστεί την διεκπεραίωση των υπηρεσιών διαδικτύου (Web Services),

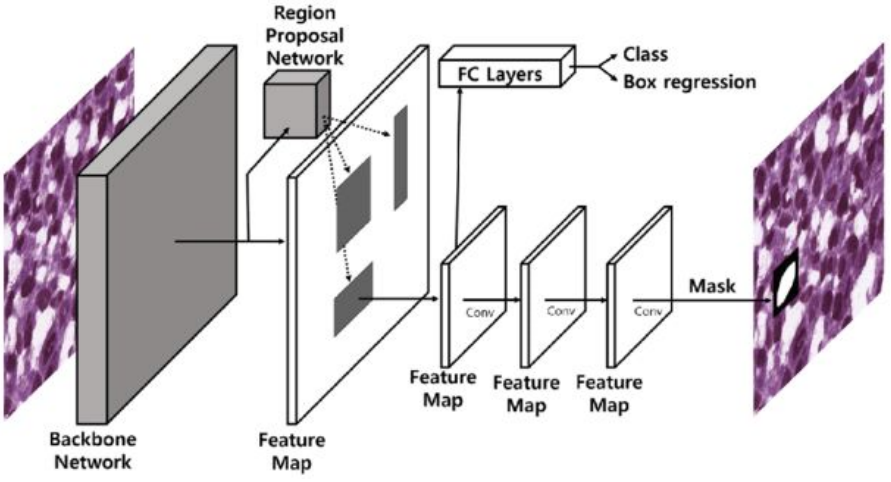
- Η Διεθνοποίηση (I18N) και τοπικοποίηση (L10N) είναι εύκολο για να υλοποιηθούν και να χρησιμοποιηθούν από τους προγραμματιστές του λογισμικού, ώστε να κάνουν την εφαρμογή παγκόσμιας χρήσης σε όλες τις γλώσσες που επιθυμούν.
- Οι δυνατότητες που δίνει το Yii για την προσωρινή μνήμη (cache) είναι ένα από τα βασικά πλεονεκτήματα. Η προσωρινή μνήμη βοηθάει την μείωση του κόστους των υπολογισμών των δεδομένων, καθώς αυτά είναι ήδη αποθηκευμένα στην κεντρική μνήμη, και υποστηρίζονται σε επίπεδο βάσης δεδομένων, διαδικτυακής ιστοσελίδας και δυναμικού περιχυμένου.
- Ένα άλλο σημαντικό πλεονέκτημα του λογισμικού είναι οι διαχείριση των λαθών και αρχείων καταγράφων για τους προγραμματιστές. Με αποτέλεσμα να γίνεται εύκολη η αποσφαλμάτωση.
- Καθιστά εύκολη η δημιουργία αυτομάτων δοκίμων με χρήση των πιο γνωστών λογισμικών δοκίμων σε γλώσσα PHP, όπως το PHPUnit και το Selenium,. Η δημιουργία αυτομάτων δοκίμων είναι αναγκαία για να εξασφαλιστεί η εγκυρότητα και η ορθότητα των λειτουργικών απαιτήσεων.
- Αυτοματοποιεί τη δημιουργία κώδικα για τον σκελετό της εφαρμογής μέσο του εργαλείου Gii, το οποίο παράγει ιστοσελίδες CRUD (Create Read Update Delete). Αυτές οι ιστοσελίδες παράγονται σύμφωνα με ένα υπάρχον μοντέλο που συνδέετε άμεσά με την βάση δεδομένων

2.3 Μηχανική Όραση

Η μηχανική όραση έχει βελτιώσει τα τελευταία χρόνια σε σημαντικό βαθμό την αναγνώριση αντικειμένων και τη σημασιολογική κατάτμηση (semantic segmentation) εικόνων, ενώ παρέχει εξαγωγή αποτελεσμάτων και συμπερασμάτων σε σύντομο χρονικό διάστημα. Σε μεγάλο βαθμό, αυτή οι πρόοδος έχει επιτευχθεί από βασικά συστήματά και μεθόδους, όπως τα γρήγορα και γρηγορότερα Συνελικτικά Νευρωνικά Δίκτυα Περιοχής (Fast [5]/Faster Region Convolutional Neural Network - RCNN) [6] και τα καθολικά Συνελικτικά Δίκτυα

(Fully Convolutional Network - FCN). Οι μέθοδοι αυτές έχουν υλοποιηθεί σε βιβλιοθήκες, σε διάφορες γλώσσες προγραμματισμού, με σκοπό για την αναγνώριση των αντικειμένων και την σημασιολογικής κατάτμησης, και προσφέρουν μεγάλη ευελιξία (flexibility), στιβαρότητα (robustness), γρήγορη εκμάθηση (training) του νευρωνικού και αποτελέσματα σε σύντομο χρόνο.

Η σημασιολογική κατάτμηση είναι εν γένει μια δύσκολη διαδικασία, καθώς απαιτεί την σωστή αναγνώριση όλων των αντικειμένων σε μια εικόνα και την τμηματοποίηση της εικόνας αυτής με μεγάλη ακρίβεια στο κάθε διαφορετικό αντικείμενό. Εν ολίγης, συνδυάζει τα κλασικά στοιχεία της επεξεργασίας εικόνας και τεχνικές Μηχανικής Μάθησης στην ανίχνευση αντικειμένων, όπου ο στόχος είναι η ταξινόμησή αντικειμένων και ο εντοπισμός τους ξεχωριστά χρησιμοποιώντας ένα πλαίσιο οριοθέτησης (Bounding Box). Βασική επιδίωξη της σημασιολογικής κατάτμησης είναι η ταξινόμησή του κάθε εικονοστοιχείου σε ένα σταθερό σύνολο κατηγοριών.



Εικόνα 4: Η αρχιτεκτονική του Mask R-CNN (https://www.researchgate.net/publication/336615317)

Ίσως το πιο σημαντικό συστατικό της εφαρμογής που αναπτύχθηκε, είναι η αναγνώριση των αντικειμένων στις εικόνες και η εξαγωγή των μασκών, με σκοπό να εντοπιστούν τα ιατρικά ευρήματα. Για την διαδικασία αυτή επιλέχθηκε η μέθοδος Mask R-CNN [7]. Η μέθοδος Mask R-CNN είναι ένα βαθύ νευρωνικό δίκτυο που επεκτείνει την μέθοδος του γρήγορου Faster R-CNN, και το οποίο επιλέγεται συνήθως για την ταξινόμηση και τον εντοπισμό των πλαισίων οριοθέτησης, προσθέτοντας ένα ακόμα κλάδο για την προβλέψει μάσκας κατάτμησης σε κάθε περιοχή ενδιαφέροντος (Region of Interest - RoI). Στην ουσία ο κλάδος της μάσκας είναι ένα FCN που εφαρμόζεται σε κάθε RoI, προβλέποντας την μάσκα της

κατάτμησης εικονοστοιχείο προς εικονοστοιχείο. Η ευέλικτη αρχιτεκτονική καθιστά την μέθοδο Mask R-CNN εύκολη στην υλοποίηση και γρήγορη στον εντοπισμό των αντικειμένων, προσθέτοντας απλώς μια μικρή υπολογιστή επιβάρυνση.

Εξορισμού η μέθοδος Mask R-CNN είναι μια διαισθητική επέκταση του γρηγορότερου R-CNN που προστέθηκε για την κατασκευή της μάσκας εντοπισμού. Η μέθοδος του γρηγορότερου R-CNN δεν υλοποιήθηκε για επεξεργασία των εικόνων σε επίπεδο εικονοστοιχείου, αλλά με την χρήση συγκεκριμένων τεχνικών (όπως η RoIPool), επιχειρεί την χονδροειδή χωρική κβαντοποίηση της εικόνας σε τετράγωνες περιοχές για την εξαγωγή των χαρακτηριστικών. Έτσι επιστρέφει προτάσεις περιοχών αλλά κλάση, η οποίες ωστόσο για κάθε σημείο αγκύρωσης (Anchor), είναι χονδροειδώς εντοπισμένες και γραμμικές [8]. Για την βελτίωση του φαινομένου αυτού η τεχνική Mask R-CNN χρησιμοποιεί πιο ευέλικτες τεχνικές (όπως η RoIAlign) με σκοπό την πιστή κατάτμηση των χωρικών τοποθεσιών σε οτιδήποτε σχήμα. Πρακτικά η μέθοδος αποτελείται από δυο στάδια, στο πρώτο στάδιο προτείνει περιοχές που ενδέχεται να βρίσκονται τα αντικείμενα ενδιαφέροντάς στην εικόνα μέσω ενός αλλού μικρού νευρωνικού δικτύου (Region Proposal Network - RPN), ενώ σε δεύτερο στάδιο, βρίσκει την κλάση του αντικείμενό, το πλαίσιο οριοθέτησης, και τέλος την μάσκα του αντικειμένου σε επίπεδο εικονοστοιχείου.

Τα δυο στάδια λειτουργίας του Mask R-CNN συνδέονται άμεσα και με την ραχοκοκαλιά (Backbone) του Νευρωνικού Δικτύου. Η ραχοκοκαλιά είναι ένα βαθύ νευρωνικό δίκτυο με πυραμιδική μορφή χαρακτηριστικών (Feature Pyramid Network [9]), το οποίο αποτελείται από ένα από κάτω-προς-τα-πάνω (bottom-up) και ένα από πάνω-προς-τα-κάτω (top-bottom) μονοπάτι με πλευρικές συνδέσεις. Το μονοπάτι από κάτω-προς-τα-πάνω είναι ένα συνελικτικό νευρωνικό δίκτυο για την εξαγωγή χαρακτηριστικών (όπως για παράδειγμα το ResNet ή τα VGG δίκτυα). Είναι σημαντικό να σημειωθεί ότι τα δίκτυα πυραμιδικής μορφής πλεονεκτούν έναντι των άλλων συνελικτικών νευρωνικών δικτύων, κυρίως διότι διατηρούν ισχυρά σημασιολογικά χαρακτηριστικά σε διάφορες κλίμακες ανάλυσης.

1.4 Γλώσσες Προγραμματισμού

Δεδομένου του όγκου και της πολυπλοκότητας της εφαρμογής, οι γλώσσες προγραμματισμού που χρησιμοποιηθήκαν είναι πολλές και η κάθε μια διαχειρίζεται το επιμέρους διεργασίες. Παρακάτω δίδεται μια μικρή περιγραφή για κάθε μια από τις γλώσσες

που χρησιμοποιήθηκαν στο πλαίσιο της εργασίας.

1.4.1 Η γλώσσα Python

Η γλώσσα Python είναι μια διερμηνευομένη γλώσσα, υψηλού επιπέδου και γενικού σκοπού, απλή, δυναμική, αποδοτική, εύκολη στην εκμάθηση και επεκτάσιμη γλώσσα προγραμματισμού, όπως περιγράφεται από τους δημιουργούς και τους χρήστες της. Είναι κατάλληλη για όλων των επίπεδων προγραμματιστών, από τον πιο αρχάριο ως τον πιο εμπείρου. Μπορεί να χρησιμοποιηθεί τόσο για εκπαιδευτικούς σκοπούς όσο και για την ανάπτυξη ολοκληρωμένων και περίπλοκων εφαρμογών όπως για παράδειγμα εφαρμογών Τεχνίτης Νοημοσύνης. Στο πλαίσιο αυτό υπάρχουν υλοποιήσεις με τις οποίες δύναται η χρήση και η παραμετροποίηση της μεθόδου Mask R-CNN.

Εικόνα 5: Λογότυπο της Python (https://www.python.org/)

Επιπροσθέτως, η γλώσσα Python διαθέτει αποδοτικές δομές δεδομένων υψηλού επίπεδου και υποστηρίζει μια αρκετά αποτελεσματική προσέγγιση στον αντικειμενοστραφή προγραμματισμό. Χρησιμοποίει την τεχνολογία συλλογής απορριμμάτων (Garbage Collection [10]), η οποία προσφέρει αυτόματη διαχείριση μνήμης και καθιστώντας την υλοποίηση εύκολη, καθώς δεν απαιτείται από τους χρήστες να διαχειριστούν τις μεταβλητές που θα διαγραφούν από την μνήμη τους συστήματος. Υποστηρίζει και τις άλλες δυο γνωστές προγραμματιστικές προσεγγίσεις, την συναρτησιακή και την διαδικαστική. Η φιλοσοφίας του σχεδιασμού της Python δίνει έμφαση στην αναγνωσιμότητα και την εύκολη συγγραφή του κώδικα. Είναι διερμηνευομένη γλώσσα προγραμματισμού και μπορεί να χρησιμοποιηθεί τόσο για την δημιουργία σεναρίων εντολών όσο και για την γρήγορη ανάπτυξη ολοκληρωμένων εφαρμογών σε όλα τα γνωστά λειτουργικά συστήματα στην αγοράς (όπως Linux, Windows, MacOS κτλ.).

Η γλώσσα προγραμματισμού Python διαθέτει πληθώρα ολοκληρωμένων και ετοίμων βιβλιοθηκών που μπορούν να χρησιμοποιηθούν ευκολά και άμεσα. Επίσης παρέχει την

δυνατότητα στον πηγαίο κώδικα να ενωθεί με τμήματα σε C, C++ και να εκτελείται γρηγορά και ομαλά. Τα προγράμματα σε Python είναι συμπαγή, ευανάγνωστα και η συγγραφή τους είναι εύκολη και ταχύτερη σε άλλες δημοφιλείς γλώσσες προγραμματισμού όπως η Java, στην οποία η σύνταξη της την καθιστά δυσανάγνωστη και δύσκολη στην εκμάθηση και συγγραφή. Επίσης, ο κώδικα της μπορεί να ομαδοποιηθεί σε μονάδες (modules) και σε πακέτα (packages), στο ίδιο ακριβώς πλαίσιο που χρησιμοποιείται από όλες τις γνωστές γλώσσες προγραμματισμού για την ομαδοποίηση των λειτουργιών ή για την δημιουργία βιβλιοθηκών.

1.4.2 Η γλώσσα προγραμματισμού PHP

Η γλώσσα PHP (PHP: Hypertext Processor) [11] είναι γενικού σκοπού και σεναρίων γλώσσα προγραμματισμού σχεδιασμένη για την δημιουργία δυναμικών διαδικτυακών ιστοσελίδων. Δημιουργήθηκε αρχικά από το τον προγραμματιστή Rasmus Lerdorf το 1994 αλλά τώρα υποστηρίζεται και υλοποιείται από ομάδα προγραμματιστών .Το περιεχόμενο της περνάει για επεξεργασία από ένα συμβατό διακομιστή όπως Apache HTTP Server και συνήθως πλαισιώνετε από κώδικα HMTL για την εμφάνιση και μορφοποίηση των αποτελεσμάτων. Τα σενάρια PHP (PHP scripts) κατά βάση περιέχουν κώδικα HTML, JavaScript και CSS και εκτελούνται στον φυλλομετρητή (Browser), σε σύγκριση με την πηγαία PHP που πρώτα εκτελείται στον διακομιστή και έπειτα από την μεταγλώττιση της εκτελείται. Η γλώσσα δύναται να εκτελεστεί σε όλα γνωστά λειτουργικά συστήματα όπως Linux,Windows, MacOS, και προσθέτει σημαντικές δυνατότητες όπως η παρακολούθηση πρόσβασης σε πραγματικό χρόνο, η επεξεργασία και δημιουργία εικόνων, η σύνδεση με απομακρυσμένους υπολογιστές και πολλά αλλά.

Εικόνα 6: Λογότυπο της PHP (https://www.php.net)

Επιπλέον, η γλώσσα PHP είναι και αυτή διερμηνευομένη και ανοιχτού κώδικα γλώσσα, η οποία είναι υλοποιημένη σε μονάδες (module), ενώ παρέχει και διεργασία στο σύστημα

υποστήριξης του λογισμικού (π.χ. Common Gateway Interface (CGI)) που δίνει την δυνατότητα συνδέσεις με όλες τις γνωστές βάσεις δεδομένων για την παραγωγή δυναμικού και προσωποποιημένου περιεχομένου. Αυτή την στιγμή μονοπωλεί σχεδόν το ποσοστό των ιστοσελίδων που βρίσκονται στο διαδίκτυο διότι είναι ευρέως χρησιμοποιουμένη γλώσσα, με τεραστία κοινότητά (Community).

Τα πλεονεκτήματα της, γλώσσας είναι ότι α) διαθέτει υψηλή απόδοσή και είναι εύκολο να εκτελεστεί από τον διακομιστή με λίγου πόρους, δηλαδή το κόστος για την λειτουργία είναι χαμηλό β) η διασύνδεση με τις βάσεις δεδομένων είναι εύκολη και γρήγορη. Οι παραπάνω είναι οι δυο βασικές αίτιες για τις οποίες η PHP χρησιμοποιήθηκε για την υλοποίηση του συστήματος υποστήριξης (Backend), καθώς επίσης επειδή επιτυγχάνονται υψηλές επιδόσεις με χαμηλό χρόνο συντήρησης. Η γλώσσα είναι εύκολη στην εκμάθηση και την υλοποίηση εφαρμογών, δεδομένου ότι είναι εύκολη στην κατανόηση και χρήση και βασίζεται κυρίως στις γλώσσες C και Perl. Για τις λιγότερο σημαντικές και διεργασίες όπως η αποστολή email και ειδοποιήσεων, αυτές υλοποιούνται με λίγες γραμμές κώδικα αφού οι βιβλιοθήκες είναι ενσωματωμένες και εύκολες στην χρήση.

1.4.3 Η γλώσσα προγραμματισμού JavaScript

Η γλώσσα προγραμματισμού JavaScript είναι μια υψηλού επιπέδου, δυναμική, διερμηνευόμενη γλώσσα, η οποία υποστηρίζει δέσμες ενεργειών και ακολουθεί πιστά το πρότυπο ECMAScript. Αυτό σημαίνει ότι οι εντολές της διερμηνεύονται από τον διερμηνέα (από την μηχανή περιβάλλοντος JavaScript όπως το V8 [12]). Η μηχανή περιβάλλοντος παρέχει κανόνες για την πρόσβαση στην κυριά μνήμη, για την αλληλεπίδραση με το λειτουργικό σύστημα καθώς επίσης ελέγχει τη σύνταξη της γλώσσας αν είναι έγκυρη. Αυτό δίνει την δυνατότητα της επέκτασης ενός προγράμματος κατά την εκτέλεση του. Για παράδειγμα ο διακομιστής μπορεί να στείλει ένα ασύγχρονα τμήμα κώδικα στο περιηγητής και αυτός να το εκτελέσει εκείνη την χρονική στιγμή. Η δυνατότατά αυτή κάνει την JavaScript ιδανική για εφαρμογές ενιαίας προβολής, όπως ο επισημειωτής εικόνας (Image Annotator) που υλοποιήθηκε ο οποίος φορτώνει δυναμικά εμφανίζει και σχεδιάζει εικόνες και μάσκες στον χρήστη.

Η γλώσσα JavaScript [13] δημιουργήθηκε αρχικά από τον Bendan Eich της εταιρείας Netscape. Εισήχθη το 1995 ως τρόπος προσθήκης προγραμμάτων στις ιστοσελίδες στο πρόγραμμα περιήγησης Netscape Navigator. Αρχικά η γλώσσα ονομαζόταν Mocha αλλά

άλλαξε αρκετές ονομασίες μέχρι να καταλήξει στο εμπορικό όνομα «JavaScript». Η επιλογή της ονομασίας δεν είναι τυχαία και έγινε για σκοπούς μάρκετινγκ, λόγω της γλώσσας προγραμματισμού Java, που ήταν δημοφιλής εκείνη την χρονολογία. Έτσι δανείστηκε το ίδιο όνομα ώστε να προσελκυσθούν προγραμματιστές στην εκμάθηση της. Ωστόσο, οι δυο αυτές γλώσσες προγραμματισμού πέραν του ονόματος δεν έχουν άλλα κοινά μεταξύ τους, η σύνταξη είναι διαφορετική και οι χρήσεις τους είναι για διαφορετικούς σκοπούς εφαρμογών.

Εικόνα 7: Λογότυπο της ECMA, ECMAScript πρότυπο(http://www.ecma-international.org/)

Κάποια επιπλέον χαρακτηριστικά που θα πρέπει να αναφερθούν αφορούν στο γεγονός ότι η JavaScript είναι αντικειμενοστραφής γλώσσα, αντιμετωπίζει τα πάντα ως αντικείμενα και βασίζεται σε πρότυπα (Prototype-based programming). Η αντικειμενοστραφής τεχνική βασισμένη στα πρότυπα, αξιοποιεί ένα αντικείμενό το οποίο μπορεί να χρησιμοποιηθεί ως πρότυπο ενός αλλού αντικειμένου ώστε να επιτευχθεί η κληρονομικότητα όλων των μεθόδων και ιδιοτήτων, από το ένα αντικείμενό στο άλλο. Το σημαντικότερο πλεονέκτημα αυτής της Τεχνικής είναι το γεγονός ότι επιτρέπει την κλωνοποίηση και την επέκταση του κώδικα εύκολα και γρηγορά. Η γλώσσα υποστηρίζει περισσότερα από ένα πρότυπα προγραμματισμού (Multi-Paradigm) και επιτρέπει στους προγραμματιστείς να συγγράψουν με οποιοδήποτε στυλ προγραμματισμού θέλουν, το οποίο κάνει την εκμάθηση και συγγραφή προγραμμάτων εύκολη και γρήγορη.

1.5 HTML

Η HTML είναι μια γλώσσα Σήμανσης Υπερκειμένου (Hypertext Markup Language) που χρησιμοποιείται για την δημιουργία ιστοσελίδων. Τις ιστοσελίδες έπειτα μπορεί να προσπελάσει οποιοσδήποτε έχει πρόσβαση στο διαδίκτυο μέσω ενός περιηγητή/ φυλλομετρητή, αφού ο φυλλομετράς λαμβάνει τα HTML αρχεία από τον διακομιστή η τοπικά και ερμηνεύει τα αρχεία αυτά σε γραφικά στον χρήστη. Συνήθως, συνδυάζεται από

άλλες τεχνολογίες για την μορφοποίηση των επάλληλων φύλλων ύφους (Cascading Style Sheets) και για την λειτουργικότητα χρησιμοποιούνται σεναριακές γλώσσες όπως η JavaScript.

Εικόνα 8: Λογότυπο της HTML5

Η γλώσσας HTML Δημιουργήθηκε από έναν υπάλληλο του CERN, τον Tim Berners-Lee , που πρότεινέ και έθεσε το πρωτότυπο INQUIRE, ένα σύστημα για τους ερευνητές του οργανισμού να χρησιμοποιούν και να μοιράζονται έγγραφα. Το 1989, έγραψε ένα σημείωμα που προτείνει ένα σύστημα υπερκείμενου βασισμένο στο διαδίκτυο. Ο Berners-Lee καθόρισε την γλώσσα HMTL και έγγραψε ένα πρόγραμμα περιήγησης και το λογισμικού διακομιστή που θα εκτελούσε η ιστοσελίδα. Ένα χρόνο αργότερα, η πρώτη έκδοση της HTML έγινε διαθέσιμη στο κοινό με την ονομασία “HTML Tags”, και από τότε η γλώσσα αυτή έγινε η βασική γλώσσα του διαδικτύου.

Τέλος, τα HTML στοιχεία είναι δομικά υλικά για την δημιουργία σελίδων. Με διάφορες δομές όπως εικόνες και αλλά αντικείμενα όπως διασπαστικές φόρμες μπορούν να προτεθούν σε μια σελίδα. Η HTML παρέχει μέσα για την δημιουργία δομημένων εγγράφων, υποδηλώνοντας την δομική σημασιολογία για τον κείμενο όπως λίστες, συνδέσμους, εικόνες, παραγράφους και αλλά στοιχεία. Τα στοιχεία αυτά οροθετούνται με ετικέτες και μπορούν να προστεθούν το ένα μέσα στο άλλο για να τις ανάγκες του προγράμματος. Η εισαγωγή των ετικετών μέσα σε μια σελίδα μπορείς να γινεί σε δυο φάσεις. Κατά την πρώτη φάση ο χρήστης θα ανοίξεις την ιστοσελίδας και θα δει το περιεχόμενό, ενώ στην δεύτερή φράση, ανάλογά τις ανάγκες της εφαρμογής, μέσω JavaScript μπορούν να γίνουν ασύγχρονα ή σύγχρονα, η εισαγωγή καινούριων ετικετών μέσα στην σελίδα και να εμφανιστούν άμεσα στον φυλλομετρητή του πελάτη.

1.6 Επάλληλα Φύλλα Ύφους (CSS)

Τα επάλληλα Φύλλα ύφους (Cascading Style Sheets - CSS) είναι μια στιλιστική γλώσσα που διευκολύνει τον τρόπο με τον οποίο θα διατάσσονται και εμφανίζονται τα διαφορά στοιχεία ιστοσελίδων σε γλώσσες σήμανσης όπως η HTML. Η γλώσσα CSS αποτελεί και η αυτή αναπόσπαστο κομμάτι στην δημιουργία όλου του διαδικτύου μαζί με HMTL και JavaScript, όπως έχει αναφερθεί και ανωτέρω. Έχει σχεδιαστεί για τον διαχωρισμό της εμφάνισης του περιεχομένου, συμπεριλαμβανομένης της διάταξη, των χρωμάτων και τις γραμματοσειρές. Αυτός ο διαχωρισμός μπορεί να βελτιώσει την προσβασιμότητα και την εμφάνιση του περιεχομένου, να παρέχει περισσότερή ευελιξία και έλεγχο στην προδιαγραφή των χαρακτηριστικών, ενώ επιτρέπει σε πολλές ιστοσελίδες να μοιράζονται την μορφοποίηση. Αυτό έχεις ως αποτέλεσμα να μειώσει την πολυπλοκότητα και την επανάληψη στο δομικό περιεχόμενό καθώς το αρχείο CSS μπορείς να αποθηκευτεί στην προσωρινή μνήμη (Cache memory) και να βελτιώσει την ταχύτητα πρόσβασης των σελίδων που μοιράζονται αρχείο αυτό.

Όσων αφορά τη σύνταξη της γλώσσας αυτή είναι απλή, χρησιμοποιεί διάφορες Αγγλικές λέξεις για να καθορίσει τα ονόματα διαφορών στιλιστικών ιδιοτήτων, και αποτελείται από τον επιλογέα (Selector) ο οποίος στην ουσία αναφέρει το επιθυμητό κομμάτι της HTML στο οποίο θέλουμε να κάνουμε αλλαγές, ακολουθούμενο από διάφορες δηλώσεις. Οι δηλώσεις μπορεί να είναι μια ή παραπάνω και κάθε μια αποτελεί μια ιδιότητα μαζί με την τιμή που επιθυμεί ο προγραμματιστής να της εισάγει για τις ανάγκες της εφαρμογής.

1.7 Διαχείριση Βάσεων Δεδομένων (MySQL)

Η MySQL [14] είναι το σύστημα διαχείρισης σχεσιακών βάσεων δεδομένων (Relational Database Management System) που σχεδιαστικέ και υλοποιήθηκε για την αποθήκευση δεδομένων όλης της εφαρμογής μας. Η σχεσιακή βάση οργανώνει τα δεδομένα σε έναν η πολλούς πίνακες δεομένων, στους οποίους οι τύποι δεδομένων μπορείς να σχετίζονται μεταξύ τους. Αυτές οι σχέσεις βοηθούν στην δομή των δεδομένων και είναι το βασικό χαρακτηριστικό που κάνει το σύστημα αυτό να λέγεται σχεσιακό.

Εικόνα 9: Λογότυπο της MySQL(https://www.mysql.com/)

Η γλώσσα που χρησιμοποιείται από τους προγραμματιστές για την επικοινωνία με το σύστημα MySQL, είναι η Δομημένη Γλώσσα Ερωτημάτων (SQL). Η γλώσσα SQL μπορεί να χρησιμοποιηθεί για την εισαγωγή, αναζήτηση, ενημέρωση και διαγραφή των δεδομένων στην βάση. Η γλώσσα αυτή επιτρέπει την δημιουργία πινάκων για την εισαγωγή των δεδομένων στην αρχή της εφαρμογής αλλά ακόμα και όταν η εφαρμογή είναι ολοκληρωμένη και εκτελείται. Επίσης, η SQL δύναται να πραγματοποιήσει άλλες λειτουργίες όπως την βελτιστοποίηση και συντήρηση της βάσης δεδομένων, γεγονός που καθιστά την δουλειά των προγραμματιστών πιο εύκολη και ανώδυνη.

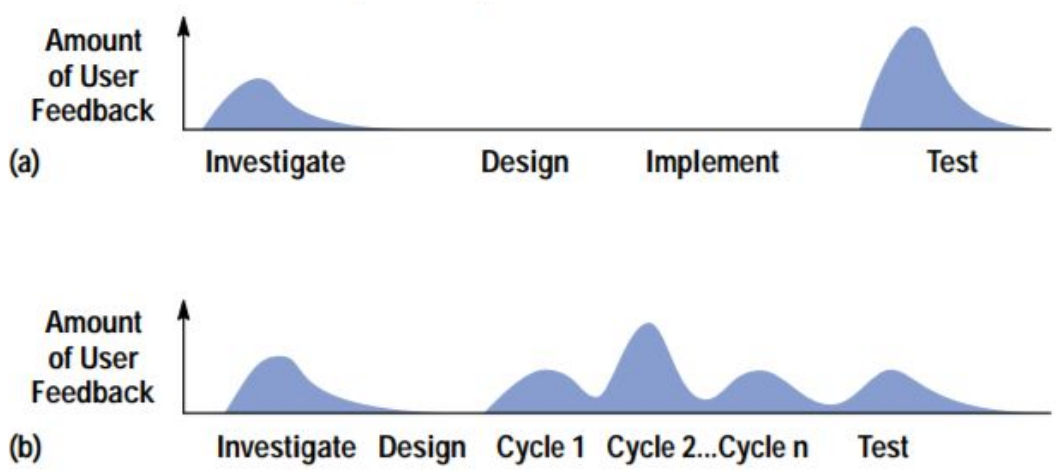
Τέλος θα αναφέρουμε μερικά πλεονεκτήματα του συστήματος MySQL [15]. Το μεγαλύτερο πλεονέκτημα του συστήματος είναι η ταχύτητα προσπέλασης των δεδομένων καθώς υποστηρίζει την ικανότητα για πολλαπλά νήματα (Multithreading). Πρακτικά δηλαδή μπορούν πολλαπλοί χρήστες ταυτόχρονα να προσπελάζουν τα ιδιά ή διαφορετικά δεδομένα. Ωστόσο, αυτό που κάνει τη MySQL ευρέως γνωστή και χρησιμοποιούμενη αφού είναι η ευκολία της χρήσης, η ασφάλεια και το γεγονός ότι είναι συμβατή με πολλά λειτουργικά συστήματα όπως Linux, Windows κτλ. Επίσης ένα άλλο πλεονεκτήματα πηγάζει από το γεγονός ότι διατηρείται το μέγεθος της μνήμης που χρησιμοποίει χαμηλό.

Κεφάλαιο 2

2. Ανάλυση και σχεδιασμός αρχιτεκτονικής

Στα πλαίσια της διπλωματικής εργασίας αναπτύχθηκε η πλατφόρμα υπολογιστικού νέφους με σκοπό την αναγνώριση και σήμανση ιστολογικών εικόνων. Οι χρήστες της εφαρμογής θα έχουν την δυνατότητα να ανεβάσουν εικόνες στην πλατφόρμα με σκοπό την αυτόματη αναγνώριση των αντικειμένων και μασκών.

Στο κεφάλαιο αυτό θα αναλυθούν με λεπτομέρεια τα πρώτα στάδιά της ανάπτυξης λογισμικού. Σημειώνεται ότι στη καταγραφή των απαιτήσεων του χρήστη, ο χρήστης είναι ένα φανταστικός χαρακτήρας που εποικήθηκε για την εξυπηρέτηση και διεκπεραίωση του έργου. Η ανάλυση και ο σχεδιασμός του λογισμικού είναι η πρώτη φάσης της ανάπτυξης του. Η ανάπτυξη λογισμικού είναι μια πολύπλοκη και χρονοβόρα διαδικασία που ακολουθείται για την δημιουργία λογισμικού και εφαρμογών, προσφέρει βέλτιστα αποτελέσματα και κέρδος στους δυνητικού χρήστη και του προγραμματιστεί. Η μεθοδολογία που θα ακολουθηθεί στα πλαίσια της διπλωματικής, είναι η Εξελικτική Ανάπτυξη (Evolutionary Development) [16]. Η διαδικασία αυτή έχεις ως πρώτο βήμα την καταγραφή των μη λειτουργικών απαιτήσεων, ενώ η συγγραφή των λειτουργικών απαιτήσεων γίνεται σταδιακά ανά κομμάτι της εφαρμογής. Έπειτα ακολουθεί η ανάπτυξη λογισμικού ανά κομμάτι και παρουσιάζεται ώστε να επιβεβαιωθεί από τον χρήστη ότι είναι σύμφωνα με τις απαιτήσεις του. Με την διαδικασία αυτή ελαχιστοποιείται το ρίσκο να αποτύχει η εφαρμογή γιατί σε κάθε βήμα ο χρήστης είναι εκεί και εγκρίνει. Η προσέγγιση που χρησιμοποιήθηκε υπερτερεί σε σύγκριση με την μέθοδο του Καταρράκτη κατά την οποία δεν υπάρχει αλληλεπίδραση με τον χρήστη παρά μόνο όταν αναπτυχθεί όλο το λογισμικό.



Εικόνα 10: Επικοινωνία με τον πελάτη στην μεθοδολογία του α) Καταρράκτη και b) Εξελικτική Ανάπτυξη

2.1 Αναλυτική καταγραφή των μη λειτουργικών απαιτήσεων

Η αναλυτική καταγραφή των μη λειτουργικών απαιτήσεων πριν την ανάπτυξη λογισμικού είναι αναγκαία διαδικασία λόγω του γεγονότος ότι οι απαιτήσεις αυτές αφορούν την άμεση λειτουργικότητα της εφαρμογής. Δηλαδή ορίζουν τους περιορισμούς στις υπηρεσίες ή στις λειτουργίες που προφέρει το σύστημα. Συνήθως αφορούν την αξιοπιστία, τις τεχνολογίες και τα πρότυπα που πρέπει να χρησιμοποιηθούν κατά το στάδιο ανάπτυξη της εφαρμογής, ωστόσο σε πολλές περιπτώσεις ο διαχωρισμός των δυο κατηγοριών απαιτήσεων αποτελείται από μια λεπτή γραμμή. Οι μη λειτουργικές απαιτήσεις που καταγράφηκαν σύμφωνα με τις επιθυμίες του χρήστη είναι:

2.1.1 Αξιοπιστία

- Το σύστημα θα πρέπει να είναι πάντα διαθέσιμο για λειτουργία της πλατφόρμας, όταν το απαιτεί ο χρηστής
- Το λογισμικό θα πρέπει να εγγυάται στο χρήστη ότι θα συμβαίνουν μηδαμινές αποτυχίες όταν το λογισμικό εκτελείται από αυτόν.
- Το σύστημα θα πρέπει να δείχνει τα ακαριαία και χωρίς αλλοίωσή των αποτελεσμάτων του Mask R-CNN, που είναι η έξοδος από το νευρωτικού δίκτυού
- Η εφαρμογή θα πρέπει να αποθηκεύει τα δεδομένα που θα εισάγονται από τον χρήστη και σε καμία περίπτωση δεν θα πρέπει οποιοσδήποτε να μπορεί να τα

διαγράψει χωρίς την συγκατάθεση του

2.1.2 Απόδοση

- Η εφαρμογή θα διαθέτει φιλικό περιβάλλον προς τον τελικό χρήστη. Η κυριά οθόνη θα περιέχει διαγράμματα και στατιστικά για την κατάσταση της πλατφόρμας αλλά και της συνεισφοράς του.
- Στη εφαρμογής το μενού πλοήγησης και τα ονόματα των λειτουργιών των ιστοσελίδων θα πρέπει να είναι κατανοητά και διαχωρισμένα σύμφωνα με την λειτουργικότητα τους.
- Στο κομμάτι του σχεδιαστή εικόνας του λογισμικού, κάθε λειτουργικότητα πρέπει να είναι κατανοητή και σε περίπτωση που δεν την κατανοεί ο τελικός χρήστης να υπάρχει εγχειρίδιο χρήσης άμεσα προσβάσιμο.
- Στον σχεδιαστεί εικόνας της εφαρμογής, οι ενέργειες του χρηστή για την δημιουργία και την ενημέρωση των εικόνων και μασκών θα πρέπει να συγχρονίζονται αυτόματα με τον κεντρικό διακομιστή εκείνη την χρονική στιγμή.
- Η εφαρμογή θα πρέπει να διαθέτει στατιστικά των εικόνων του χρήστη που έχουμε αποθηκευτεί από το στάδιο της εξαγωγής αποτελεσμάτων του νευρωνικού δικτύου.

2.1.3 Ευχρηστία

- Η εφαρμογή θα διαθέτει φιλικό περιβάλλον προς τον τελικό χρήστη. Η κυριά οθόνη θα περιέχει διαγράμματα και στατιστικά για την κατάσταση της πλατφόρμας αλλά και της συνεισφοράς του.
- Στην εφαρμογή οι πρωτεύοντες σύνδεσμοι πλοήγησης και τα ονόματα των λειτουργιών των ιστοσελίδων θα πρέπει να είναι κατανοητά και διαχωρισμένα σύμφωνα με την λειτουργικότητα τους
- Για τον επισημειωτή εικόνας, κάθε λειτουργικότητα θα πρέπει να είναι κατανοητή και σε περίπτωση που δεν την κατανοεί ο τελικός χρήστης να υπάρχει εγχειρίδιο χρήσης άμεσα προσβάσιμο.
- Στον επισημειωτή εικόνας της εφαρμογής, οι ενέργειες του χρηστή για την δημιουργία και την ενημέρωση των εικόνων και μασκών θα πρέπει να

συγχρονίζονται αυτόματα με τον κεντρικό διακομιστή την ίδια χρονική στιγμή

- Η εφαρμογή θα πρέπει να διαθέτει στατιστικά των εικόνων του χρήστη τα οποία έχουν καταχωρηθεί από το στάδιο της εξαγωγής αποτελεσμάτων του νευρωνικού.

3.1.4 Εξελισιμότητα

- Το λογισμικό θα πρέπει να έχει την δυνατότητα να ενημερώνετε αυτόματα στο πρόγραμμα του περιηγητή του χρήστη χωρίς αυτός να εκτελεί κάποια ενέργεια
- Η εφαρμογή θα πρέπει να κρατάει όλα τα δεδομένα του χρηστή και σε περίπτωση σφαλνά δικού του θα μπορεί να τα ανακτήσει έμμεσα (επικοινωνία με τον διαχειριστή της πλατφόρμας).
- Τα λογισμικό θα πρέπει να εξελίσσεται και να εντοπίζει ορθά και αποτελεσματικά τις μάσκες των εικόνων με κάθε επανάληψης εκμάθησής του νευρωνικού δικτύου με καινούριες εικόνες σχεδιασμένες από τους χρήστες.

3.1.5 Απαιτήσεις Βάσης Δεδομένων

- Το λογισμικό θα αποθηκεύει τις δραστηριότητές των χρηστών στην βάση δεδομένων MySQL και συμβατότητα με MariaDB, και θα προσφέρει προσωποποιημένη πληροφορία ανάλογα τον ρολό και την συνεισφορά του στην πλατφόρμα
- Η εφαρμογή θα πρέπει να αποθηκεύει στο σύστημα διαχείρισης σχεσιακής βάσης (Relational Database Management System – RDBMS) τις εξόδους του νευρωνικού δικτύου και όλα τα στατιστικά που εξάγονται από την συνιστώσα της τεχνίτης νοημοσύνης.

3.1.6 Μη εγκατάσταση επιπλέον εφαρμογών

- Η εφαρμογή θα επιτρέπει στον τελικό χρήστη να την εκτελεί και λειτουργεί χωρίς καμία εγκατάσταση στην προσωπική συσκευή
- Το λογισμικό θα πρέπει να εγκατασταθεί στον κεντρικό διακομιστή από τον διαχειριστή της εφαρμογής στο λειτουργικό σύστημα Ubuntu
- Η εφαρμογή θα απαιτεί την παρουσία καρτών γραφικών Nvidia [17] με τουλάχιστον

11GB VRAM και την σωστή διασύνδεση με την πλατφόρμα παράλληλου υπολογισμού και μοντέλο διεπαφή προγραμματισμού εφαρμογών (API) για κάρτες γραφικών Nvidia (CUDA [18])

2.1.7 Συμβατότητα

- Το λογισμικό του χρήστη θα πρέπει να είναι συμβατό με τα προγράμματα πλοήγησης Chrome, Edge, Opera και Firefox, και η εμφάνιση των διεπαφών χρήστη να είναι ακριβώς η ίδια.

3.1.8 Ασφάλεια

- Η εφαρμογή θα πρέπει να αυθεντικοποιεί τον χρήστη με τους κωδικούς προσβάσεις που έχει στην κατοχή του και μόνο τότε να εμφανίσει τις προσωποποιημένες πληροφορίες του
- Το λογισμικό θα πρέπει να στιβαρό και άτρωτό σε κακόβουλες διαδικτυακές επίθεσής (Hacking).
- Στο διακομιστή που είναι εγκαταστημένη η εφαρμογή θα πρέπει να έχει μόνο πρόσβαση ο διαχειριστής του συστήματος και η διασύνδεση του θα γίνεται με χρήση ασφαλών λογισμικών απομακρυσμένης πρόσβασης.

3.1.9 Λειτουργία

- Το λογισμικό θα πρέπει να είναι διαθέσιμο κάθε ώρα και στιγμή για τον τελικό χρηστή.
- Η εφαρμογή θα επιτρέπει την διαδικασία εκπαίδευσης και εξαγωγής αποτελεσμάτων από το νευρωνικού δίκτυο σειριακά, δηλαδή μια λειτουργία κάθε φορά με την σειρά εισαγωγής.

Σύμφωνα με τις παραπάνω μη λειτουργικές απαιτήσεις προκύπτει ότι η υλοποίηση θα διεξαχθεί σε δυο σκέλη α) ένα σκέλος θα αφορά το σύστημα υποστήριξης της εφαρμογής (Backend) και β) το δεύτερο σκέλος θα αφορά την συνιστώσα της τεχνητής νοημοσύνης (Artificial Intelligence App). Το σύστημα υποστήριξης θα είναι εγκαταστημένο στο κεντρικό

διακομιστή του συστήματος και θα παρέχει 24ωρη εξυπηρέτηση στον πελάτη, και η εκτέλεση θα πρέπει να είναι συμβατή με όλους τους γνωστούς φυλλομέτρησες και τα λειτουργικά συστήματα. Η συνιστώσα τεχνητής νοημοσύνης θα είναι εγκατεστημένη σε άλλη υποδομή με υψηλή υπολογιστική ισχύ και κάρτες γραφικών με σκοπό την εκπαίδευση, την επικύρωση και τον έλεγχο των συνελικτικών δικτύων. Η διαδικασία εκπαίδευσης και η εξαγωγή των αποτελεσμάτων του Συνελικτικού Νευρωνικού Δικτύου θα πρέπει να γίνεται στις κάρτες γραφικών για λογούς απόδοσης και ως εκ τούτου η αποστολή των δεδομένων θα αποστέλλονται απευθείας στον κεντρικό διακομιστή μέσω ασφαλούς δίαυλου.

2.2 Καταγραφή των λειτουργικών απαιτήσεων

Οι λειτουργικές απαιτήσεις είναι οι απαιτήσεις του χρήστη που αφορούν άμεσα τις λειτουργικότητες, δυνατότητες της πλατφόρμας, την απόδοση και την εμφάνιση της. Σύμφωνα με την διαδικασία της εξελικτικής ανάπτυξης που ακολουθείται, αρχικά καταγράφονται οι πιο σημαντικές απαιτήσεις του χρήστη για το σύστημα και έπειτα διεξάγεται ο πρώτός κύκλος υλοποίησης. Να σημειωθεί ότι λόγω των τεράστιων και περίπλοκων απαιτήσεων του χρήστη στο τρέχων κεφάλαιο οι κύκλοι απαιτήσεων είναι δυο και η εφαρμογή διαχωρίστηκε σε τρία διαφορετικά μέρη α) σύστημα υποστήριξης (Backend), β) Επισημειωτής Εικόνας (Image Annotator) και συνιστώσα τεχνητής νοημοσύνης (Artificial Intelligence app). Και για λογούς κατανόησης του αναγνώστη στο κεφάλαιο θα παρουσιαστούν οι λειτουργικές απαιτήσεις σύμφωνα με το κάθε ξεχωριστό μέρος της εφαρμογές που ανήκει η συγκεκριμένη απαίτηση.

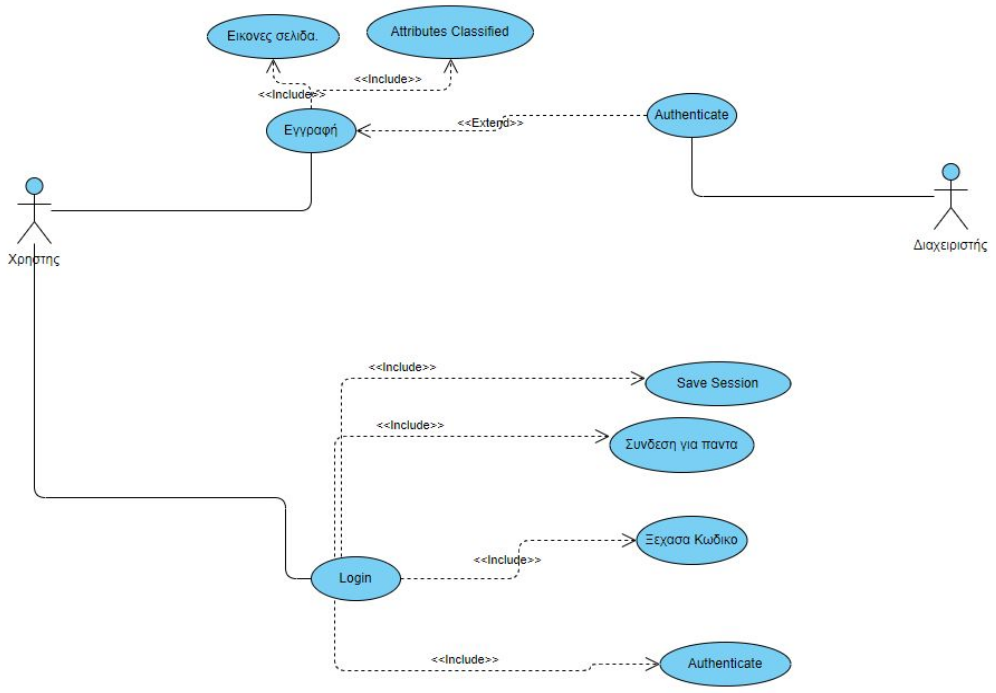
3.2.1 Λειτουργικές απαιτήσεις και σχεδιασμός Συστήματος Υποστήριξης (Backend)

Αναφέρουμε ότι το σύστημα υποστήριξης είναι το πρώτο μέρος της πλατφόρμας, στο οποίο ο τελικός χρήστη θα έχει την δυνατότητα να διαχειρίζεται και να βλέπει όλη την εφαρμογή μέσα από το πρόγραμμα του περιηγητή. Οι κύριες λειτουργικές απαιτήσεις που καταγράφηκαν σύμφωνα με τις επιθυμίες του χρήστη έχουν ως εξής:

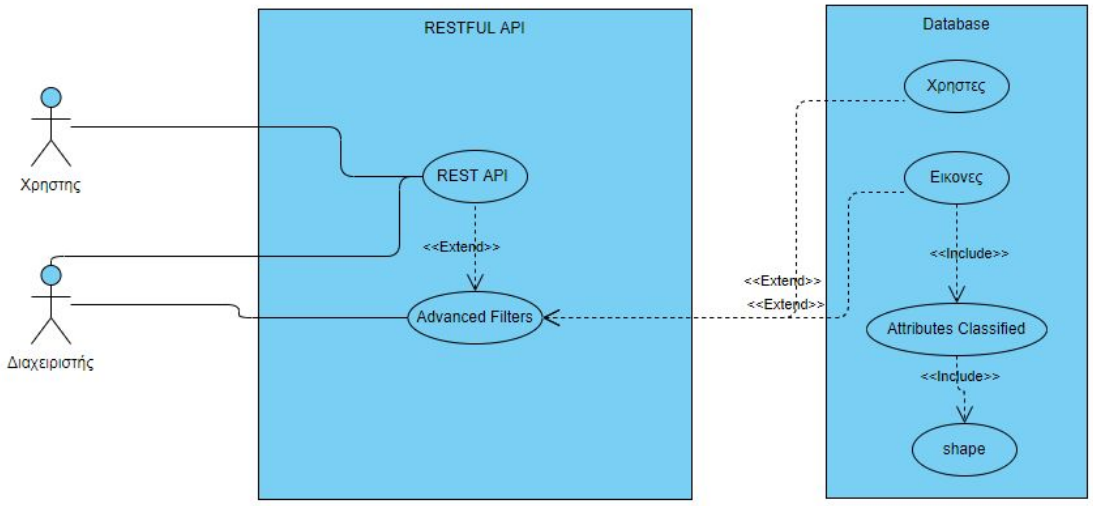
- Το λογισμικό θα πρέπει να υποστηρίζει δυο οντότητές χρηστών Διαχειριστές (Admins) και Ειδικούς Ιατρούς (Doctors). Οι διαχειριστές θα έχουν πλήρης

διαχείριση της πλατφόρμας, ενώ οι ιατροί θα έχουν περιορισμένης πρόσβαση και δεν θα πρέπει να μπορούν περιηγηθούν στις οθόνες των διαχειριστών. Θα έχουν ωστόσο την δυνατότητα να βλέπουν περιεχόμενό από συγκεκριμένους χρήστης εφόσον τους έχει καταχωρηθεί το δικαίωμα

- Η εφαρμογή θα επιβεβαιώνει τους κωδικούς του χρήστη και έπειτα θα εισέρχεται στην πλατφόρμα. Σε περίπτωση που έχει χάσει η ξεχάσει τους κωδικούς αυθεντικοποίηση θα του δίνεται η δυνατότητα επανάκτησης (Εικόνα 11)
- Το λογισμικό θα περιέχει μια κεντρική οθόνη (Dashboard), στην οποία θα ενσωματώνεται γραφήματα για την πρόοδο της εφαρμογής αλλά και συνεισφορά του τελικού χρήστη
- Η εφαρμογή θα πρέπει να διαχωριστεί σύμφωνα με τους τελικούς χρήστες που έχουν κάνει τις ενέργειες
- Η πλατφόρμα θα επιτρέπει την κληρονομικότητα του μοντέλου βαρών (Weight Model) το οποίο θα περιέχει τα βάρη που θα αποθηκεύονται κατά την εκπαίδευση του νευρωνικού δικτύου
- Το σύστημα θα επιτρέπει την δημιουργία συνόλου εικόνων για την εκπαίδευση (training), την επικύρωση (validation) και την τελική ποσοτικοποίηση (quantification) νέων εικόνων.
- Το σύστημα θα επιτρέπει στον χρήστη να ανεβάζει εικόνες και σε περίπτωση λάθους, θα εμφανιστεί το κατάλληλο μήνυμα
- Η εφαρμογή θα πρέπει να περιέχει Restful API (Representational State Transfer) για την διασύνδεση με άλλες πλατφόρμες και σκέλη της εφαρμογής. Τα δεδομένα αυτά θα προκύπτουν από την διασύνδεση της πλατφόρμας με την βάση δεδομένων όπως φαίνεται στο διάγραμμα περίπτωσης χρήσης (Use Case diagram) της εικόνας 12.



Εικόνα 11:Use Case για την σύνδεση του τελικού πελάτη στην πλατφόρμα



Εικόνα 12: Use Case των τελικών χρηστών που κάνουν αιτήσεις στο API και η διασύνδεση του με την βάση δεδομένων

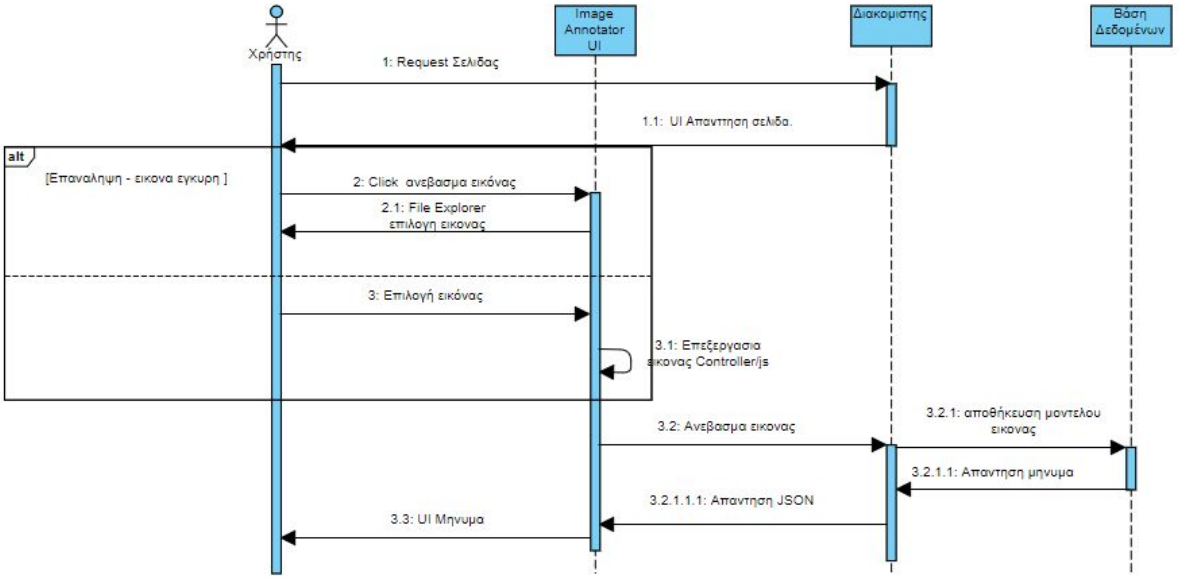
3.2.2 Λειτουργικές απαιτήσεις και σχεδιασμός του Επισημειωτή (Annotator)

Ο επισημειωτής εικόνας αποτελεί το υποσύστημα της πλατφόρμας, το οποίο είναι υπεύθυνο για εμφάνιση των εικόνων και των μασκών που προέρχονται από το αποτέλεσμα της εξόδου του υποσυστήματος της Τεχνητής Νοημοσύνης. Επίσης, στο υποσύστημα αυτό θα γίνεται η

αποτύπωση, σχεδίαση των μασκών στις εικόνες που επιθυμεί ο χρήστης είτε να τις εισάγει στο σύνολο εκπαίδευσης είτε να τις διορθώσει. Οι βασικότερες λειτουργικές απατήσεις που καταγραφτήκαν από τον χρήστη είναι οι έξης:

- Η εφαρμογή θα δίνει την δυνατότητα στον τελικό χρήστη να την χρησιμοποιήσει είτε μέσω ενσωμάτωσής στο σύστημα υποστήριξης είτε αυτόνομη τοπικά στον υπολογιστή. Η διασύνδεση, οι αιτήσεις (Request) και η λειτουργικότητα θα γίνεται μέσω API στο σύστημα υποστήριξης.
- Το λογισμικό θα δίνει την δυνατότητα στους διαχειριστές να βλέπουν όλη την δραστηριότητα της εφαρμογής και στους ιατρούς την δική τους δραστηριότητά ή κάποιου αλλού ιατρού άμα του έχει δοθεί το δικαίωμα.
- Το σύστημα θα επιτρέπει στους τελικούς χρηστές να χρησιμοποιούν την εφαρμογή σαν εφαρμογή μοναδικής όψης (Single View Application), δηλαδή χωρίς να γίνεται επαναφόρτωση της ιστοσελίδας από το πρόγραμμα περιήγησης.
- Η εφαρμογή θα επιτρέπει στους τελικούς χρήστες να προσθέτουν και να διαχειρίζουν πολλές εικόνες ταυτόχρονα. Αφού γινεί η διαδικασία αυτή η πλατφόρμα θα τους ενημερώνει με το κατάλληλο μήνυμα
- Το λογισμικό θα πρέπει να περιέχει την σχεδίαση σχημάτων (μάσκες), τα σχήματα αυτά θα είναι κύκλος, έλλειψη, ορθογώνιο – τετράγωνο, πολύγωνο και σχεδίαση με το χέρι (free drawing ή multipolygon).
- Η πλατφόρμα θα αποθηκεύει αυτόματα όλες τις αλλαγές του χρηστή και χωρίς αυτός να κάνει κάποια ενέργεια
- Το σύστημα θα επιτρέπει την μεγέθυνση/σμίκρυνση (zoom in/out) στις εικόνες και επιλογή προ υπάρχουσας μάσκας για ενημέρωση ή διαγραφή

Στην παρακάτω Εικόνα 13 εμφανίζεται ένα σενάριο εισαγωγής νέας εικόνας βιοψίας από το χρήστη.



Εικόνα 13:Διάγραμμά ακολουθίας UML για την προσθήκη νέας εικόνας από τον τελικό χρήστη.

3.2.3 Λειτουργικές απαιτήσεις και σχεδιασμός για το Artificial Intelligence App

Η εφαρμογή Τεχνητής Νοημοσύνης είναι το υποσύστημα της πλατφόρμας, το οποίο αναλαμβάνει την αυτόματη αναγνώριση των αντικειμένων, σύμφωνα με την εμπειρία με τις εκπαιδεύσεις που του έχει δοθεί από τις εικόνες και μάσκες των τελικών χρηστών. Επομένως το υποσύστημα είναι ο σκελετός/κορμός της πλατφόρμας και τα αλλά δυο υποσυστήματα (Backend, Image Annotator) είναι υποστηρικτικά. Οι πιο σημαντικές λειτουργικές απαιτήσεις που καταγραφτήκαν από τον πελάτη είναι:

- Το λογισμικό θα είναι αυτόνομο από την πλατφόρμα και η διασύνδεση του με το Σύστημα υποστήριξης θα γίνεται μέσω API
- Το πρόγραμμα θα δίνει την δυνατότητα στον τελικό πελάτη να μπορείς να εκπαιδεύει το νευρωνικό δίκτυο σύμφωνα με τις ρυθμίσεις, εικόνες και μάσκες που θα έχει εισάγει στο σύστημα υποστήριξης και στην βάση δεδομένων
- Η εφαρμογή θα πρέπει να μπορεί να αναγνωρίσει αντικείμενα και μάσκες στις εικόνες που θα ανατεθούν από τους τελικούς χρήστες μέσω αρχείων προηγουμένων εκπαιδεύσεων (Αρχεία Βαρών Νευρωνικού Δικτύου)
- Το λογισμικό θα διατηρεί τα προηγούμενα αρχεία εκπαιδεύσεων και θα επιτρέπει την κληρονομικότητα μεταξύ τους, ώστε το σύστημα να γινεί πιο έγκυρο και

αποτελεσματικό στα αποτελέσματα του νευρωνικού δικτύου

- Το σύστημα θα πρέπει να είναι πλήρως παραλληλοποιημένο και ικανό να εκτελείται στις κάρτες γραφικών μέσω CUDA διεπαφή για όλες τις λειτουργίες του. Έτσι οι διαδικασίες εξαγωγής αποτελεσμάτων (και ειδικά της εκπαίδευσης) να μην επιβαρύνουν χρονικά και υπολογιστικά από την κεντρική μονάδα επεξεργασίας.
- Το λογισμικό θα πρέπει να αποθηκεύει τα αποτελέσματα (μάσκες ,ρυθμίσεις, στατιστικά και αντικείμενα) της εξόδου του νευρωνικού στο κεντρικό διακομιστή, ώστε να γινεί η αξιολόγηση τους από τον τελικό χρήστη

Κεφάλαιο 3

3. Υλοποίηση Πλατφόρμας

Στο πλαίσιο της παρούσας διπλωματικής αναπτύχθηκε η πλατφόρμα υπολογιστικού νέφους για την αναγνώριση και ποσοτικοποίηση ιστολογικών εικόνων. Όπως αναφέρθηκε και στο προηγούμενο κεφάλαιο, λόγο της πολυπλοκότητας και των απαιτήσεων της πλατφόρμας, το σύστημα διαχωρίστηκε στα τρία επιμέρους υποσυστήματα (Backend, Image Annotator και Artificial Intelligence app).

Στο κεφάλαιο αυτό θα αναλυθεί ο πηγαίος κώδικας, οι βιβλιοθήκες που χρησιμοποιήθηκαν και οι γραφικές του χρήστη διεπαφές που υλοποιήθηκαν για να γίνουν κατανοητές οι λειτουργίες και οι διαδικασίες με τις οποίες λειτουργεί η εφαρμογή. Τα τρία υποσυστήματα της εφαρμογής που θα αναλύσουμε καλύπτουν όλες λειτουργικές απαιτήσεις που αναφέρθηκαν ανωτέρω αλλά και άλλες απαιτήσεις που δεν αναφέρονται στην παρούσα διπλωματική. Ωστόσο η πλατφόρμα που υλοποιήθηκε καλύπτει και όλες τις μη λειτουργικές απαιτήσεις που σημειωθήκαν από τον χρήστη.

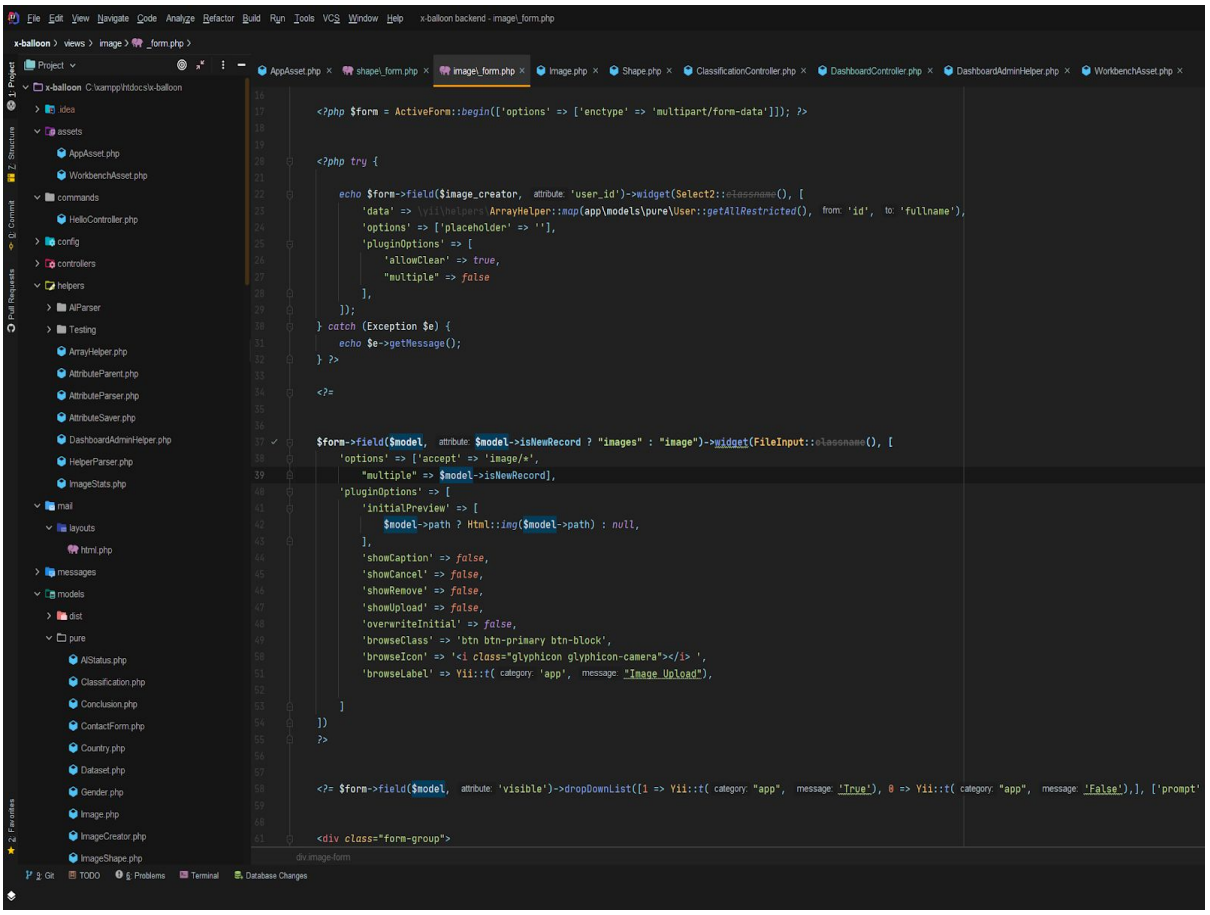
3.1 Σύστημα Υποστήριξης (Backend)

Το σύστημα υποστήριξης είναι το σύστημα το οποίο ευθύνεται για την ολοκλήρωση της πλατφόρμας με τα αλλά δυο υποσυστήματος, καθώς και για την ταυτοποίηση και την επιδείξει προσωποποιημένης πληροφορίας στους χρήστες. Η σημαντικότερή λειτουργία αφορά την δυνατότητα στους χρηστές να δημιουργούν δικά τους σύνολα εκπαίδευσης και επικύρωσης τα οποία θα αποτελούνται από μάσκες και εικόνες δίκες τους για την εκπαίδευση και εξαγωγής αποτελεσμάτων του νευρωνικού δικτύου.

Οι τεχνολογίες που χρησιμοποιήθηκαν για την υλοποίησης του συστήματος υποστήριξης είναι, η αρχιτεκτονική MVC για την δομή των αρχείων αλλά και την μεθοδολογία ανάπτυξής πανό στο Yii 2 Framework στην γλώσσα προγραμματισμού PHP. Για την διαχείριση των

πακέτων και βιβλιοθηκών χρησιμοποιήθηκε ο Composer που είναι και υπεύθυνος για την εγκατάσταση όλων των αναγκαίων πακέτων για την λειτουργία της εφαρμογής. Για τις γραφικές διεπαφές του τελικού πελάτη χρησιμοποιήθηκε η βιβλιοθήκη Bootstrap [19], και για την αναπαράσταση και μορφοποίηση των γραφικών στοιχείων η προγραμματιστική γλώσσα JavaScript και η στιλιστική γλώσσα CSS. Επίσης για την αναπαράσταση των γραφημάτων χρησιμοποιήθηκε η ανοιχτού λογισμικού βιβλιοθήκη Morris [20], ώστε τελικός χρήστης να βλέπει όλη την πληροφορία της συγκεκριμένης διεπαφής με μια ματιά και να κατανοεί σε βάθος την κατάσταση των δεδομένων του. Ωστόσο, για την καλύτερη εμπειρία στην αλληλεπίδραση του τελικού χρηστή με τις γραφικές διεπαφές χρησιμοποιήθηκε η βιβλιοθήκη JQuery [21], για την ασύγχρονη εκτέλεση και φόρτωση δεδομένων και γραφικών στοιχείων αλλά και για την διαχείριση των συμβάντων των ενεργειών του χρηστή στα συγκεκριμένα γραφικά στοιχεία.

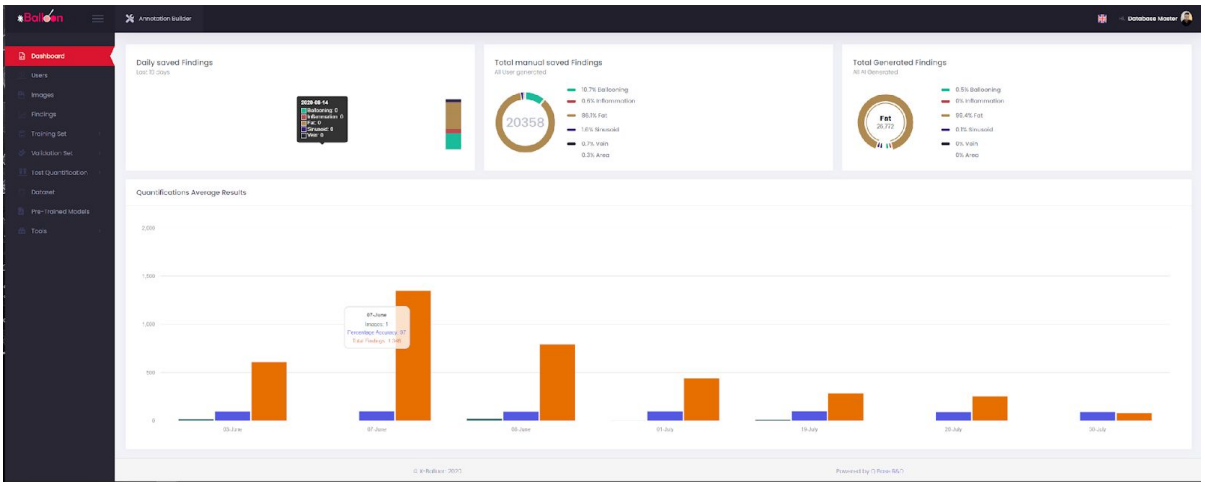
Επιπλέον, για την αποθήκευση όλων των δεδομένων και την εξαγωγή τους, χρησιμοποιήθηκε το σύστημα διαχείρισης σχεσιακών βάσεων δεδομένων MySQL. Η βάση δεδομένων συνδέεται με το σύστημα υποστήριξης μέσω τις ενσωματωμένης βιβλιοθήκης της PHP και η διαχείριση και αποτύπωση όλων των πληροφορίων γίνεται άμεσα και εύκολα μέσω του Framework Yii. Όπως αναφέρεται και ανωτέρω το Yii προσφέρει αυτόματη μεταγλώττιση του σχεσιακού μοντέλου σε αντικειμενοστραφή, με σκοπό να καθίσταται δυνατή η διαχείριση των μοντέλων ως γενικού τύπου αντικείμενα. Η υφιστάμενη ιδιότητα του Yii είναι πλήρως επαναλαμβανομένη, και μειώνει τον χρόνο υλοποίησης των μοντέλων διευκολύνοντας τον προγραμματιστή. Στην Εικόνα 14 παρουσιάζεται μια μερική δομή του συστήματος υποστήριξης και συγκεκριμένα η υλοποίηση της φόρμας εισαγωγής των εικόνων.



Εικόνα 14: Μερική δομή του πηγαίου κώδικα στο Backend και η φόρμα εισαγωγής εικόνων

3.1.1 Dashboard

Η γραφική ιστοσελίδα Dashboard παρουσιάζει ένα στιγμιότυπο της προόδου όλης της πλατφόρμας και στους δυο τύπου χρηστών. Στους ειδικούς ιατρούς εμφανίζεται και η συνεισφορά στη σχεδίαση μασκών που έχει γινεί από τους ιδίους. Πιο συγκεκριμένα, η σελίδα εμφανίζει γραφήματα με το πλήθος των μασκών και εικόνων που εισάγονται από τους χρήστες της εφαρμογής σύμφωνα με την ημερολογιακή ημέρα (τελευταίες 16 μέρες). Παρουσιάζεται η εμφάνιση ξεχωριστών γραφημάτων του πλήθους όλων μασκών που έχουν εισαχθεί από τους χρήστες και από την έξοδο του νευρωνικού δικτύου. Ένα επιπλέον γράφημα που θα προσφέρεται στους τελικούς χρήστες είναι το μέσο ποσοστό επιτυχίας εικόνων και μασκών που είναι αποτέλεσμα του υποσυστήματος τεχνητής νοημοσύνης και η εισαγωγή πραγματοποιείται μέσω API στο σύστημα υποστήριξης. Το αποτέλεσμα της υλοποίησης και η εμφάνιση του Dashboard με τις λειτουργίες που αναφέραμε φαίνεται στην Εικόνα 15.



Εικόνα 15: Εμφάνιση του Dashboard για τους τελικούς χρήστες.

Επιπλέον, για να γίνει κατανοητή στον αναγνώστη η λογική υλοποίησής των παραπάνω γραφημάτων θα αναλυθεί η ο πηγαίος κώδικας της κλάσης DashboardAdminHelper εικόνας 16. Η κλάση αυτή είναι βοηθητική και χρησιμοποιείται από τον DashboardController. Είναι υπεύθυνη για την εξαγωγή δεόμενων από την βάση μέσω των αντικειμένων μοντέλων και η εξαγωγή της πραγματοποιείται σε μορφή κατανοητή από την γλωσσά προγραμματισμού JavaScript και τη βιβλιοθήκη Morris. Η συνάρτηση getShapeClassesGroupByDateJSON περιέχει την λειτουργικότητα για τη μέτρηση του πλήθους των μασκών που εχουν εισαχθεί από τους χρηστές τις τελευταίες ημέρες. Αρχικά γίνεται εξαγωγή δεδομένων από την βάση μέσω ερωτήματος (Query) SQL από το αντικειμενοστραφές μοντέλο Shape, το ερώτημα αυτό περιέχει την φυσική ένωση shape και image_shape, για την αντιστοίχιση των εικόνων με τις μάσκες τους και η ομαδοποίηση γίνεται σύμφωνα με την ημέρα εισαγωγής της μάσκας. Το αποτέλεσμα του ερωτήματος αυτού είναι αντικείμενα τύπου Shapes με όλες τις μεταβλητές. Έπειτα τα αντικείμενα αυτά εισάγονται σε επαναληπτικές διαδικασίες ώστε να μορφοποιηθεί σωστά το αποτέλεσμα για να μπορεί να γίνει σε μορφή JSON (JavaScript Object Notation). Η μορφή JSON προσφέρει ευκολία στην ανταλλαγή πληροφοριών από τις προγραμματιστές σε γλώσσες PHP και JavaScript στην παρούσα εφαρμογή.

```php
<?php

namespace app\helpers;

use app\models\pure\Classification;

class DashboardAdminHelper
{
    public static function getShapeClassesGroupByDateJSON()
    {
        $models = \app\models\pure\Shape::findBySql("
select count(*) as id,cast(shape.date_created as DATE ) as
date_created,shape.class_id,shape.shape_type_id from shape
     inner join image_shape on image_shape.shape_id=shape.id and deleted=0
and testing_id is null
    where shape.date_created >   DATE_SUB(CURDATE(), INTERVAL 15 DAY)
    group by CAST(date_created AS DATE),class_id
    order by date_created ")->all();
        $result = [];
        foreach ($models as $model) {

            if (!isset($result[$model->class_id]))
                $result[$model->class_id] = [];
            $result[$model->class_id] [] = ["count" => $model->id, "date" =>
$model->date_created];
        }
        $final_array = [];
        foreach ($result as $key => $model) {
            $classification = Classification::findOne($key);

            $final_array[$classification->name]["values"] = $model;
            $final_array[$classification->name]["pretty_name"] =
$classification->pretty_name;
            $final_array[$classification->name]["color"] = $classification-
>color;
        }
        return json_encode($final_array);
    }
    public static function getShapeClassesJSON()
    {
        $models = \app\models\pure\Shape::findBySql("select count(*) as
id,shape.class_id as date_created,shape.class_id,shape.shape_type_id
       from shape
     inner join image_shape on image_shape.shape_id=shape.id and deleted=0
and testing_id is null
    group by class_id")->all();
        $result = [];
        $classifications = Classification::find()->all();

        foreach ($classifications as $classification) {
            $result[$classification->id] = array_merge($classification-
>getAttributes(), ["count" => 0]);
        }
        foreach ($models as $model) {


            $result[$model->class_id] = array_merge(
```

Εικόνα 16: Πηγαίος κώδικας Κλάσης DashboadAdminHelper και συνάρτησης της, την getShapeClassesGroupByDateJSON

Ακόμα μια σημαντική συνάρτηση της κλάσης DashboardAdminHelper, είναι η getPreviousConclusionByDateJSON που συλλεγεί τις μέσες τιμές όλων των αποτελεσμάτων από την έξοδο του συνελικτικού νευρωνικού δίκτυου και οι οποίες έχουν αποθηκευτεί στην βάση δεδομένων. Πιο συγκεκριμένα, πραγματοποιείται η σύνταξη του ερωτήματος SQL, το οποίο περιλαμβάνει την φυσική ένωση στους σχεσιακούς πίνακες conclusion και testing_images, ενώ η ομαδοποίηση γίνεται σύμφωνα με το αποτέλεσμα από το σύνολο ελέγχου. Ωστόσο, η έξοδος των δυο πινάκων είναι η μέση τιμή των δεδομένων του conclusion και τα αποτελέσματα εισάγονται σαν πίνακας με κάθε οντότητά του αντικειμενοστραφούς μοντέλου Conclusion. Έπειτα ο πίνακας ενημερώνεται από την επαναληπτική διαδικασία ώστε να μπορεί να τροποποιηθεί σε μορφή JSON, όπως συνέβη και στην συνάρτηση getShapeClassesGroupByDateJSON.

```
    public function getPreviousConclusionsByDateJSON()
    {
        $previousTestings = Conclusion::findBySql("select
round(avg(conclusion.average_detection)*100,2) as
average_detection,avg(conclusion.shapes) as
shapes,date(if(conclusion.date_updated is null  or conclusion.date_updated
=0,conclusion.date_created,conclusion.date_updated)) as
date_created,count(conclusion.id) as counter  from conclusion
inner join testing_images on testing_images.conclusion_id=conclusion.id
group by testing_images.testing_id
order by date_created desc
limit 30")->all();

        $result = [];
        foreach ($previousTestings as $model) {
            $result[] = ["date" => $model->date_created, "percentage" =>
$model->average_detection, "counter" => $model->counter, "shapes" => $model-
>shapes,
            ];
        }

        return json_encode($result);
    }
```

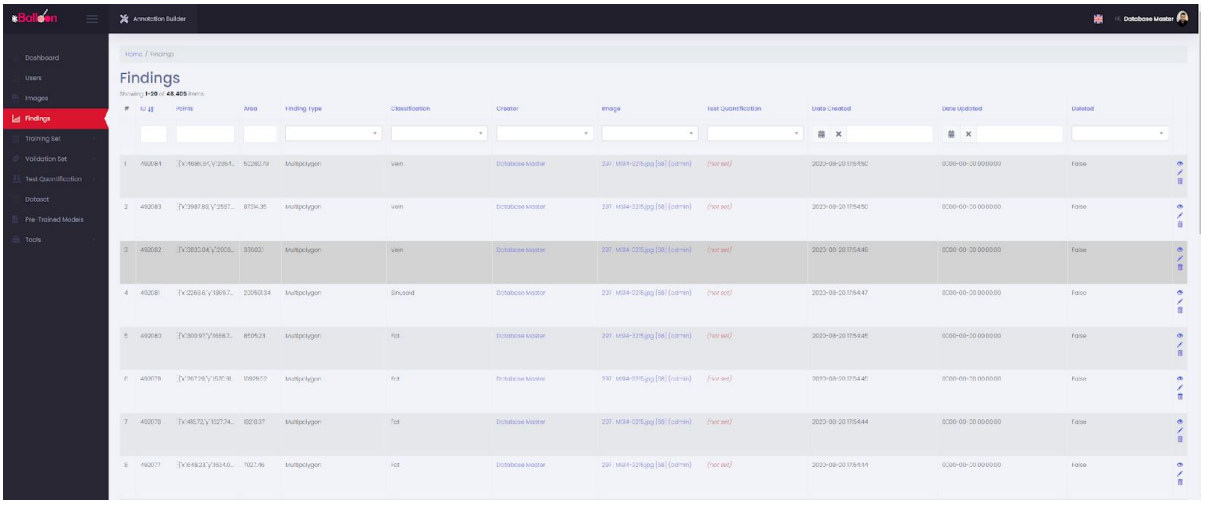
Εικόνα 17: Πηγαίος κώδικας της κλάσης DashboadAdminHelper και συνάρτησης της, την getPreviousConclusionByDateJSON

3.1.2 Shapes/Findings (Μάσκες/Ευρήματα)

Στο κεφάλαιο αυτό θα αναλύσουμε δυο πτυχές της λειτουργικότητας των ευρημάτων. Τις σελίδες που περιέχουν όψης των ευρημάτων με όλες οι πληροφορίες της συγκεκριμένης μάσκας και την όψη που περιέχει όλες τις πληροφορίες για όλες τις μάσκες (Datatable). Οι μάσκες αυτές προέρχονται από δυο πήγες, η πρώτη πηγή αφορά στην εισαγωγή από τον

χρηστή (εισαγωγή από τον επισημειωτή εικόνων) μέσω ειδικών γραφικών διεπαφών. Ενώ η δεύτερή πηγή αφορά το αποτέλεσμα το υποσύστημα τεχνητής νοημοσύνης.

Σχετικά με τον γραφικό πίνακα των ευρημάτων, η ιστοσελίδα που παράχθηκε για την προβολή και διαχείριση μασκών φαίνεται στην εικόνα 18. Ο πίνακας των ευρημάτων διαφέρει ανάλογά με τον λογαριασμού του χρήστη, καθώς για τους διαχειριστής δεν υπάρχουν περιορισμοί ενώ για τους απλούς χρηστές (Doctors) δίνεται η δυνατότητα εμφάνισης των μασκών που έχουν άμεση συσχέτιση με αυτούς. Η συσχέτιση αυτή περιλαμβάνει την εισαγωγή που έχει γινεί με δίκης του ενέργεια, από το αποτέλεσμα του υποσυστήματος τεχνητής νοημοσύνης σε εικόνα που έχουν ανεβάσει οι ίδιοι, και τέλος όλες οι μάσκες των χρηστών που τους έχουν ανατεθεί από τον διαχειριστή. Είναι εμφανές από τον εικόνα ότι δεν ανήκουν όλες οι πληροφορίες του πίνακα στην οντότητα μάσκα, αλλά είναι υπάρχουν και στοιχεία από τον συνδυασμό με άλλους σχεσιακούς πίνακες τις βάσης. Επίσης δίνεται η δυνατότητα για ασύγχρονη αναζήτηση των δεδομένων που επιθυμεί ο χρήστης και η φόρτωσή τους πραγματοποιείται χωρίς αλλαγή κατάστασης της ιστοσελίδας.



Εικόνα 18: Ιστοσελίδα με τον πίνακα όλων των ευρημάτων.

Όσων αφορά τον πηγαίο κώδικα που για την εμφάνιση του πίνακα, η πιο σημαντική συνάρτηση είναι η συνάρτηση search. Η search λαμβάνει ως παράμετρο την είσοδο αναζήτησης του χρηστή και η αρχική εμφάνιση των δεδομένων του πίνακά shapes γίνεται συμφωνά με το ID σε φθίνουσα σειρά. Τα δεδομένα που εμφανίζονται στη ιστοσελίδα του χρήστη καταγράφονται στην Select Query, και ανάλογά με τον χρήστη αν είναι ή όχι διαχειριστής, εισάγεται περιορισμός μέσω της φυσικής ένωσης με τον πίνακα shape και shape_creator. Αυτό ο περιορισμός περιέχει και την αναγκαία σύνδεση του χρήστη με την

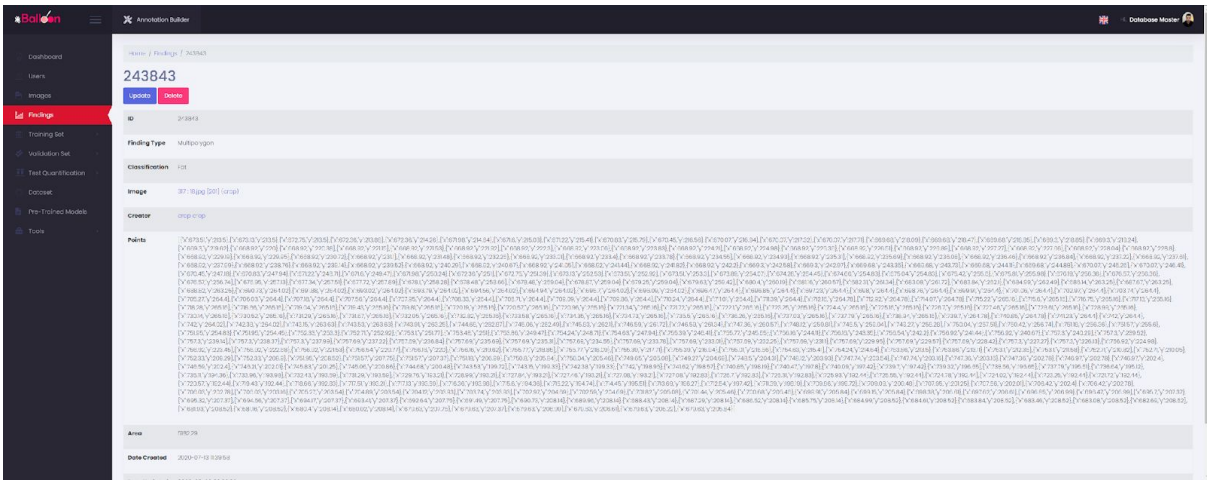
ιδιοκτησία της μάσκας. Έπειτα δίνεται η δυνατότητα άλλων χαρακτηριστικών του πίνακα που δεν συσχετίζονται άμεσα με την οντότητα shape, αλλά αυτό συμβαίνει έμμεσα μέσω φυσικής ένωσης. Ακολουθεί η διαδικασία κατά την οποία απαιτείται η εμφάνιση τον δεδομένων της αναζήτησης του χρηστή στον πίνακα, και είναι η ευθύνη της συνάρτησής andFilterWhere του αντικειμένου. Αυτή λαμβάνει ως είσοδο την μεταβλητή που πίνακά και την είσοδο του χρήστη (εφόσον υπάρχει) και πραγματοποιεί την διαδικασία ισότητας. Το αποτέλεσμα του αντικειμένου query είναι μια πρόταση SQL για την βάση δεδομένων,

```
public function search($params)
{
    $query = Shape::find()->select([
        "shape.*",
        "image_shape.image_id as image_id",
        "image_shape.deleted as shape_deleted",
        "shape_creator.user_id as shape_creator",
        "image_shape.testing_id as testing_id"
    ]);
    $creator_appender = "";
    if(Yii::$app->user->identity->isManager)
        $creator_appender = "and
        (shape creator.user id=".Yii::$app->user->id." or exists(select
user_manager.user_id  from user_manager where
shape creator.user id=user manager.user id and
user_manager.manager_id=".Yii::$app->user->id."))";

    $query->innerJoin("image_shape", "image_shape.shape_id=shape.id");
    $query->innerJoin("shape_creator", "shape_creator.shape_id=shape.id
$creator_appender");
    // add conditions that should always apply here
    $dataProvider = new ActiveDataProvider([
        'query' => $query,
        'sort'=> ['defaultOrder' => ['id' => SORT_DESC]],
    ]);
    $dataProvider->sort->attributes["image id"] = [
        'asc' => ['image_shape.image_id' => SORT_ASC],
        'desc' => ['image shape.image id' => SORT DESC],
    ];
    $dataProvider->sort->attributes["testing_id"] = [
        'asc' => ['image_shape.testing_id' => SORT_ASC],
        'desc' => ['image_shape.testing_id' => SORT_DESC],
    ];
    $dataProvider->sort->attributes["shape_deleted"] = [
        'asc' => ['image_shape.deleted' => SORT_ASC],
        'desc' => ['image_shape.deleted' => SORT_DESC],
    ];
    $dataProvider->sort->attributes["shape_creator"] = [
        'asc' => ['shape_creator.user_id' => SORT_ASC],
        'desc' => ['shape_creator.user_id' => SORT_DESC],
    ];
    $this->load($params);

    if (!$this->validate()) {
        return $dataProvider;
    }
    // grid filtering conditions
    $query->andFilterWhere([
        'id' => $this->id,
        'shape_type_id' => $this->shape_type_id,
        'testing_id' => $this->testing_id,
        'class_id' => $this->class_id,
        'image_id' => $this->image_id,
        'image_shape.deleted' => $this->shape_deleted,
        'user_id' => $this->shape_creator,

    ]);
```

Εικόνα 19:Συναρτηση Search, για την εμφάνιση και αναζήτηση των ευρημάτων

Σχετικά με την ιστοσελίδα που είναι υπεύθυνη για την συνολική εμφάνιση ενός συγκεκριμένου αντικειμένου Shape, παρουσιάζεται ένα στιγμιότυπο στην Εικόνα 20. Στην ιστοσελίδα αυτή ισχύει ο περιορισμός για την δυνατότητα προβολής ανάλογα με τον τύπο λογαριασμού χρήστη ή ανάθεσης από διαχειριστή. Το πιο σημαντικό χαρακτηριστικό αυτής

της προβολής είναι οι συντεταγμένες (points). Στην μεταβλητή points αποθηκεύονται οι συνταραγμένες του δεδομένων σε μορφή json και σε περίπτωση που η μάσκα αυτή είναι τύπου multipolygon τα δεδομένα αυτά είναι αριθμητικά πολλά. Άλλα χαρακτηριστικά που παρουσιάζονται είναι το εμβαδόν, ο δημιουργός, η κλάση ταξινόμησης, η εικόνα που ανήκει και ημερομηνία εισαγωγής.



Εικόνα 20: Ιστοσελίδα View ευρήματος.

Όσον αφορά τον πηγαίο κώδικα της προβολής αυτής, το πιο σημαντικό σκέλος είναι το DetailView, το οποίο είναι Widget στο ενσωματωμένο πακέτο του Yii. Το widget αυτό δίνει την δυνατότατά εισαγωγής του αντικειμενοστραφούς μοντέλου καθώς και τα χαρακτηριστικά που θέλει ο προγραμματιστής να εμφανίσει. Για παράδειγμα στην Εικόνα 21, το widget λαμβάνει ως είσοδο ένα πίνακα (array) που περιέχει το σχήμα (Shape) και ένα από τα χαρακτηριστικά του το οποίο δεν είναι άμεσα συνδεδεμένο, είναι το id της κλάσεις (class_id). Το class_id είναι η κλάση που έχει ταξινομηθεί και η τιμή του περιεχομένου αυτού πραγματοποιείται με την προσπέλασης φυσικής ένωσης που συμβαίνει αυτόματα από το μοντέλο. Αντιθέτως, αυτά τα οποία είναι χαρακτηριστικά του συγκεκριμένου μοντέλου, απαιτούν απλά μια δήλωση και η εμφάνιση τους γίνεται κανονικά.

```
<?= DetailView::widget([
    'model' => $model,
    'attributes' => [
        'id',

        ["attribute" => "shape_type_id",
            "value" => function ($model) {
                return $model->shapeType->pretty_name;
            }],
        ["attribute" => "class_id",
            "value" => function ($model) {
                return $model->class->pretty_name;
            }],

        ["attribute" => "image_id",
            "value" => function ($model) {

                $image = $model->image;
                return Html::a($image->prettyName,
"./index.php?r=image/view&id=$image->id", ["target" => "_blank"]);
            },
            'format' => 'raw'],
        ["attribute" => "shape_creator",
            "value" => function ($model) {
                $user = \app\models\pure\User::findBySql("select user.*
from user
inner join shape_creator on shape_creator.user_id=user.id and
shape_id=$model->id")->one();
                return Html::a($user->fullname,
"./index.php?r=user/view&id=$user->id", ["target" => "_blank"]);
            },
            'format' => 'raw'],
        'points:ntext',
        'area',
        'date_created',
        'date_updated',
    ],
]) ?>
```

Εικόνα 21: Πηγαίος κώδικας του DetailView για την εμφάνιση πληροφοριών μάσκας.

3.1.3 Διεπαφή Προγραμματισμού Εφαρμογών (API)

Οι διεπαφές προγραμματισμού εφαρμογών (APIs) επιτρέπουν στην πλατφόρμα να συνδέεται με τα αλλά δυο υποσυστήματα, και πρέπει να σημειωθεί πως δεν περιέχουν γραφικές διεπαφές για τους χρήστες αλλά μόνο διεπαφές συνδέσεων για τους προγραμματιστές με την εφαρμογή. Για την σύνδεση στα APIs είναι ανάγκα η αυθεντικοποίηση (basic authentication) του χρήστη και ανάλογα με τον τύπο λογαριασμού που διαθέτει μπορεί να προσπελάσει διαφορετικές συναρτήσεις και δεδομένα. Στο κεφάλαιο αυτό θα αναφερθούμε στα δυο κυριότερα APIs που χωρίζονται σύμφωνα με την λειτουργικότητα που προσφέρουν στα επιμέρους υποσυστήματα.

3.1.3.1 APIs Επισημειωτή Εικόνας

Για την πλήρης και αυτόνομη λειτουργία του επισημειωτή εικόνας σαν εφαρμογή ενιαίας προβολής δημιουργήθηκε ο κεντρικός ελεγκτής (Controller) τον οποίο καλούμε WorkbenchController. Όλες οι συναρτήσεις τις κλάσης αυτής είναι ξεχωριστές διεπαφές που εκτελούν διαφορετικές λειτουργίες, άλλα όλες απαιτούν την παρουσία αυθεντικοποίησης του χρηστή (basic authentication). Μια από τις πιο σημαντικές συναρτήσεις της κλάσης αυτής είναι η actionGetShapeByImage.

```
class WorkbenchController extends Controller

{

    public $enableCsrfValidation = true;

    public $user;

public function actionGetShapesByImage()
{


    $request = Yii::$app->request->get();

    try {
        if (isset($request["id"]) && $request["id"] && ($image = $this-
>findModel($request["id"]))) {

            $testing_id = null;
            if (isset($request["testing_id"]) && $request["testing_id"])
                $testing_id = $request["testing_id"];
            $attributeParser = new AttributeParser($this->user, $image);
            $attributeParser->testing_id = $testing_id;
            return ["success" => $attributeParser->init()];

        }
    } catch (\Exception $e) {


        return ["error" => $e->getMessage()];
    }


    return ["error" => "Missing id for image id ",
    ];

}
```

Εικόνα 22: Πηγαίος κώδικας της συνάρτησης actionGetShapeByImage του ελεγκτή WorkbenchController

Η actionGetShapeByImage επιστρέφει σε JSON μορφή όλες τις μάσκες σύμφωνα με τον

χρηστή για μια συγκεκριμένη εικόνας και δέχεται ως είσοδο μέσω API (Request GET) το id της εικόνας. Εφόσον έχει δικαίωμα προσπέλασης της εικόνας φυσικά σε περίπτωση που ο χρήστης θέλει να τροποποιήσει αποτελέσματα από το νευρωνικό δικτύου, του δίνεται η δυνατότητα. Έπειτα, γίνεται η δημιουργία στιγμιότυπου του αντικείμενου AttributeParser το οποίο λαμβάνει ως είσοδο τον χρήστη και την εικόνα του χρηστή. Όπως είναι εμφανές από τον πηγαίο κώδικα της Εικόνας 23, αρχικά γίνεται η διαδικασία αναζήτησης των εγγραφών στην βάση δεδομένων που περιλαμβάνει την φυσική συνένωση των πινάκων shape και image_shape. Ενώ το αποτέλεσμα του ερωτήματος είναι ένας πίνακας από αντικείμενα shapes. Στην συνέχεια ο πίνακας εισέρχεται στην επαναληπτική διαδικασία ώστε το παραγόμενο αποτέλεσμα να είναι ένα συμβατό JSON.

Επιπλέον, σχετικά με την εισαγωγή μέσω API γίνεται η διαδικασία πολλαπλών επικυρώσεων διότι απαιτείται τα δεδομένα που εισάγονται να είναι σωστά και έγκυρα. Για παράδειγμα για την εισαγωγή ευρημάτων σε μια εικόνα (Εικόνα 24) στον πηγαίο κώδικα φαίνονται οι έλεγχοι που λαμβάνουν χώρα πριν την εκχώρηση τους στην βάση δεδομένων. Ωστόσο λόγω του γεγονότος ότι η εισαγωγή των δεδομένων δεν αφορά μόνο έναν πίνακα άλλα πολλαπλούς, χρησιμοποιείται η τεχνική της συναλλαγής (transaction). Στην περίπτωση που μια από τις εισαγωγές δεν πραγματοποιηθεί σωστά τότε όλη η συναλλαγή ανακαλείται και επιστρέφεται το κατάλληλο μήνυμα λάθους στην διεπαφή. Στην περίπτωση που όλες οι εισαγωγές γίνουν σωστά τότε επιστρέφονται όλα αντικείμενα με id ώστε να αναγνωριστούν από το επισημειωτή εικόνας και να γινεί η κατάλληλη ενημέρωση των δεδομένων αυτών σύμφωνα με την βάση.

```php
<?php

namespace app\helpers;

use app\models\pure\Shape;
use yii\helpers\Json;

class AttributeParser extends AttributeParent
{
    private $shapeResult = [];

    public $testing_id = null;

    public function __construct($user, $image)
    {
        parent::__construct($user);

        if (!($image instanceof \app\models\pure\Image))
            throw new \Exception("Image not set");
        $this->image = $image;
        $this->image_id = $image->id;

    }
    public function init()
    {
        $this->jsonResult["shapes"] = [];

        if ($this->findShapes())
            $this->normalizeToJson();
        return $this->jsonResult;


    }
    public function findShapes()
    {
        $testing_str = " and testing_id is null";
       if($this->testing_id)
           $testing_str = " and testing_id=$this->testing_id";
        $this->shapes = Shape::findBySql("select shape.* from shape
          inner join image_shape on image_shape.shape_id=shape.id and
image_id={$this->image->id} and image_shape.deleted=0 $testing_str")->all();

        return count($this->shapes) > 0;
    }

    public function normalizeToJson()
    {
        foreach ($this->shapes as $shape) {
            $attributes = $shape->getAttributes();
            array_push($this->shapeResult, $attributes);
        }
        $this->jsonResult["shapes"] = $this->shapeResult;
        return true;
    }

}
```

Εικόνα 23: Κλάση AttributeParser, δημιουργείτε στιγμιότυπο της στον WorkbenchController

```
    function saveAllModels()
    {

        $request = $this->request;
        $image = $this->image;
        $this->errors = [];
        if (isset($request["shapes"]) && is_array($request["shapes"])) {


            $shapes = $request["shapes"];

            $deleted = 0;

            foreach ($shapes as $array_shape) {

                $transaction = Yii::$app->db->beginTransaction();
                try {
                    $array_shape["points"] = is_array($array_shape["points"])
? json_encode($array_shape["points"]) : $array_shape["points"];
                    $found = false;
                    $shape = new Shape();

                    if (isset($array_shape["id"]) && $array_shape["id"]) {


                        $shape = Shape::findBySql("select shape.* from shape
inner join image_shape on image_shape.image_id=$image->id
where shape.id={$array_shape["id"]}")->one();
                        if (!$shape)
                            $shape = new Shape();
                        else {
                            $found = true;
                            if (isset($array_shape["deleted"]))
                                $deleted = intval($array_shape["deleted"]);
                        }
                    }
                    $shape->setAttributes($array_shape);

                    $shape->area = floor($shape->area * 100) / 100;

                    if (!$this->isJson($shape->points)) {
                        throw new Exception("points are not json");

                    }

//                    die(json_encode($shape->getAttributes()));

                    if ($shape->save()) {

                        if (!$found || $deleted) {

                            $this->saveImageShape($found, $deleted, $image,
$shape);

                        }
                    } else {
                        throw new \Exception($shape->getFirstErrors());
```

Εικόνα 24: Πηγαίος κώδικας για την εισαγωγή ευρημάτων σε στις εικόνες.

3.1.3.1 Artificial Intelligence app APIs

Οι πιο σημαντικές διεπαφές (APIs) που υλοποιηθήκαν για την λειτουργία του υποσυστήματος τεχνητής Νοημοσύνης της πλατφόρμας βρίσκονται στον πηγαίο κώδικα του ελεγκτή AiController. Εκεί παρέχετε όλη η λειτουργικότητα διασύνδεσης των δυο υποσυστημάτων. Όπως αναφέρθηκε στην παραπάνω παράγραφο όλα τα APIs λαμβάνουν ως είσοδο τον χρηστή και την έξοδο δεδομένων σε μορφή JSON.

```
class AiController extends \app\controllers\api\v1\BasicController
{


    public function actionGetFullDataset()
    {

        try {


            $dataset = new DatasetParser($this->user);
            return ["success" => $dataset->init(),
            ];
        } catch (\Exception $exception) {
            return ["error" => $exception->getMessage(),
            ];
        }


    }


    public function actionGetTestingSet()
    {
        try {
            $dataset = new TestingSetParser($this->user);
            return ["success" => $dataset->init()];
        } catch (\Exception $e) {
            return ["error" => $e->getMessage(),
            ];
        }

    }
```

Εικόνα 25:Πηγαίος κώδικας του AiController (μερικό κομμάτι κώδικα)

Μια από τις πιο σημαντικές λειτουργικότητες της προγραμματιστικής διεπαφή της κλάσης AiController είναι οι οδηγίες για το σύνολο δεδομένων που θα χρειαστεί το νευρωνικού δίκτυο για την εκπαίδευση. Η λειτουργικότητά αυτή συμβαίνει στην συνάρτηση actionGetFullDataset που δημιουργεί ένα στιγμιότυπο DatasetParser και επιστρέφει το αποτέλεσμα του αντικειμένου αυτού. Για την αρχικοποίηση της κλάσης αυτής συμβαίνουν τα έξης γεγονότα, α) γίνεται η αναζήτηση μέσω ερωτήματος SQL στην βάση δεδομένων που

συνδυάζει τους πίνακες dataset, validation και training, και σε περίπτωση που η κατάσταση του dataset δεν είναι ολοκληρωμένη, τότε το αποτέλεσμα της διαδικασίας καταχωρείται στο dataset και όλη οι επιμέρους πίνακες καταχωρούνται σε νέες μεταβλητές. Στη συνέχεια αναζητούνται οι εικόνες που ανήκουν στα παραπάνω σύνολα δεδομένων αλλά και οι μάσκες των εικόνων, ώστε να δημιουργηθεί ένα πίνακας JSON (συνήθως ξεπερνάει σε μεγέθους μερικά Mega Byte).

```
class DatasetParser extends HelperParser
{
public function __construct($user = null)
{

    parent::__construct($user);

    $this->datasetModel = \app\models\pure\Dataset::findBySql("select
dataset.* from dataset
         inner join validation on dataset.validation_id=validation.id
        inner join training on training.id=dataset.training_id and
training.status_id!=3 and dataset.status_id!=3
        ")->one();

    if (!$this->datasetModel) {
        throw new \Exception("dataset not found");



    }
    $this->validation = $this->datasetModel->validation;
    $this->training = $this->datasetModel->training;
    $this->trainingClassifications = $this->training-
>trainingClassifications;
    $this->trainingWeight =
\app\models\pure\TrainingWeights::findBySql("select * from training_weights
where training_id={$this->training->id}")->one();


}
```

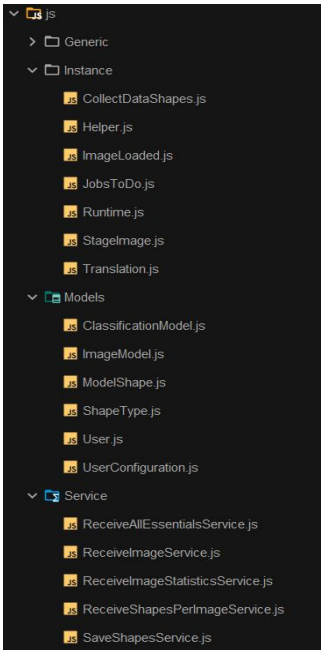
Εικόνα 26:Μερικός πηγαίος κώδικας του constructor της κλάσης DatasetParser

3.2 Επισημειωτής Εικόνας (Image Annotator)

Σε αυτό το κεφάλαιο θα αναλύσουμε τις σημαντικότερες δυνατότητες του επισημειωτή εικόνας, ο οποίος είναι υπεύθυνος για τον σχεδιασμό και την παρουσίαση των ευρημάτων στις εικόνες. Είναι η σημαντικότερη εφαρμογή για τον τελικό χρήστη γιατί του δίνεται η δυνατότητα να καταχωρεί τις μάσκες των ευρημάτων για την εκπαίδευση και να διορθώσει τα αποτελέσματα της έξοδο του συνελικτικού νευρωνικού δικτύου. Η προσπέλαση των εικόνων και των μασκών γίνεται σύμφωνα με τον ρόλο του λογαριασμού του χρήστη, αλλά και σύμφωνα με τις απαιτήσεις λογισμικού του χρήστη. Η εφαρμογή είναι όλη ενιαίας

προβολής και η φόρτωσή των ιστοσελίδων γίνεται σε ένα container

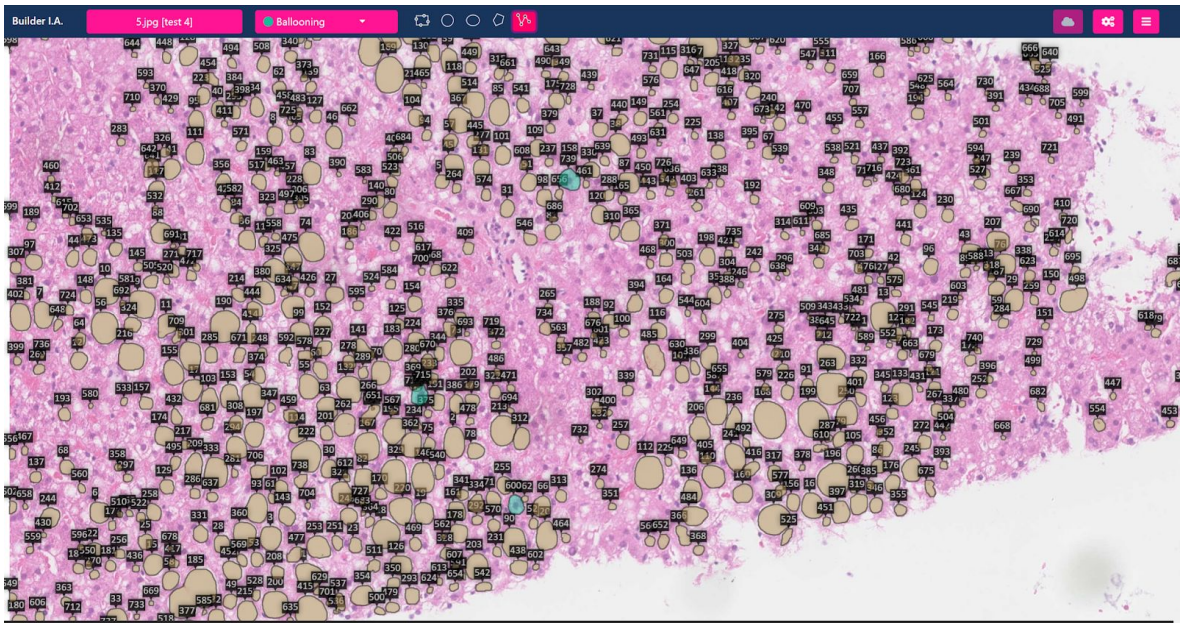
Οι τεχνολογίες που χρησιμοποιηθήκαν για την υλοποίηση του συστήματος είναι περιορισμένες καθώς όλα υλοποιήθηκαν χρησιμοποιώντας αρχιτεκτονική καθοδηγούμενη από το γεγονότα (event driven) και αντικείμενα δεδομένων. Για όλη την υλοποίηση της πλατφόρμας χρησιμοποιήθηκε μόνο η γλωσσά προγραμματισμού JavaScript και μόνο η αντικειμενοστραφής διαδικασία για την λειτουργία της πλατφόρμας. Οι βιβλιοθήκες που χρησιμοποιήθηκαν είναι ανοιχτού λογισμικού (KonvaJS), οι οποίες χρησιμοποιούνται για την αναπαράσταση σχημάτων σε καμβά (Canvas), και η βιβλιοθήκη JQuery για τα γεγονότα των ενεργειών των χρηστών που συμβαίνουν στην πλατφόρμα. Σύμφωνα με την αρχιτεκτονική και τις τεχνολογίες που χρησιμοποιήθηκαν, ο πηγαίος κώδικα διαχωρίστηκε ανάλογά με την λειτουργία που εκτελεί. Για παράδειγμα στο φάκελο με όνομα «Generic» βρίσκονται οι μητρικές κλάσεις που κληρονομούνται για την λειτουργία μιας συγκεκριμένης διαδικασίας, ενώ στον φάκελο «Service» όλα τα αντικείμενα που χρησιμοποιούνται για την επικοινωνία με το APIs του συστήματος υποστήριξης.



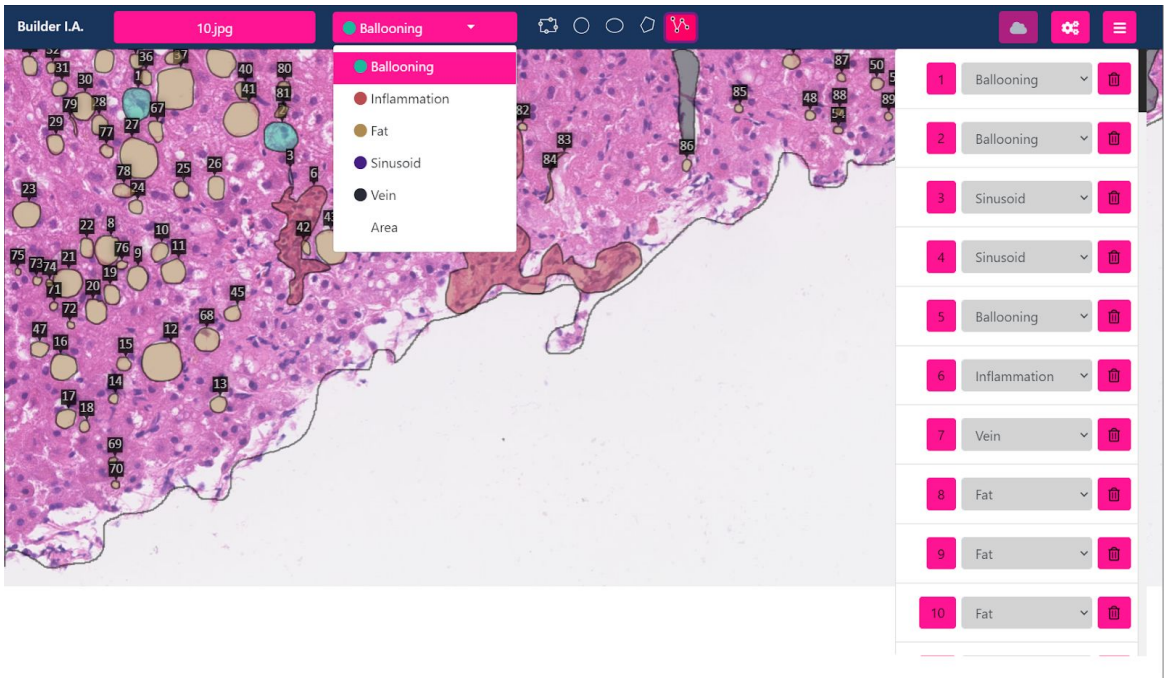
Εικόνα 27: Μερική δομή της αρχιτεκτονικής της εφαρμογής

Επιπλέον, οι σημαντικές δυνατότητες που προσφέρει η πλατφόρμα σύμφωνα με τις λειτουργικές απαιτήσεις του χρήστη είναι, η υποστήριξη πολλαπλών τύπων σχημάτων (τετράγωνο, κύκλος, έλλειψη, πολύγωνο, multipolygon/free drawing), δυνατότητα στον χρηστή να σχεδιάζει και να επεξεργάζεται όσες εικόνες επιθυμεί στην ίδια καρτέλα (tab). Η

πιο σημαντική δυνατότητα που προσφέρεται είναι η διαχείριση των μασκών στον καμβά συμφώνα με αλληλεπίδραση του χρηστή (click event) στην συγκεκριμένη μάσκα, ενώ επιτρέπει την εκτελέσει ενεργειών από τον πληκτρολόγιο. Επίσης παρέχεται η δυνατότητα δυναμικής αλλαγής στο πλήθος των κλάσεων ταξινόμησης. Τέλος, η εμφάνιση της εφαρμογής είναι λειτουργική, ανταποκρίσιμες (responsive) και εύχρηστη, ενώ όλες δυνατότητες που προσφέρονται στον χρήστη, ακόμα και όταν το πλήθος το ευρημάτων στην συγκεκριμένη εικόνα είναι πολύ μεγάλο.



Εικόνα 28: Image Annotator εικόνα με ευρήματα που προέρχονται από το αποτέλεσμα της εξόδου του νευρωνικού δικτύου.



Εικόνα 29:Image Annotator εικόνα με ευρήματα που εχουν καταχωρηθεί από τους χρήστες

4.2.1 Master.js

Η κλάση Master του επισημειωτή εικόνας έχει το μεγαλύτερο βάρος στην εφαρμογή γιατί είναι υπεύθυνη για την αρχικοποίηση των πιο σημαντικών αντικειμένων της πλατφόρμας. Επίσης δίνει την δυνατότητα στα εσωτερικά αντικείμενα να προσπελάσουν άλλα στιγμιότυπα αντικειμένων διάμεσου της κλάσης αυτής. Ωστόσο, για να γινεί κατανοητή η λειτουργικότητα και τα αίτια δημιουργίας του αντικειμένου θα εξετάσουμε τον πηγαίο κώδικα στην Εικόνα 30.

Όπως φαίνεται στον πηγαίο κώδικα του Master.js, πρώτα συμβαίνει η αρχικοποίηση των σημαντικότερων αντικειμένων που είναι αναγκαία για την λειτουργία της εφαρμογής. Όπως το αντικείμενο models που στο εσωτερικό του περιχέει πίνακες αντικειμένων για το πλήθος τον κλάσεων ταξινόμησής και τα σχήματα σχεδιασμού μασκών που θα είναι διαθέσιμα στον χρηστή. Όμως η πιο σημαντική κλάση του αντικειμένου αυτού είναι, ο πίνακας imageTabs (Map) αντικειμένων που αφορούν όλο το γραφικό περιβάλλον μέσα στο κάμβα που αλληλοεπιδρά ο χρήστης. Δηλαδή μέσα στο αντικείμενό αυτό περιέχονται τα αντικείμενα, τα συμβάντα για την τρέχουσα εικόνα και τα ευρήματα που παρουσιάζονται ή σχεδιάζονται. Το στιγμιότυπο tabHandler αφορά την μπάρα διεπαφής του χρηστή που δεν καθορίζεται από την τρέχουσα σχεδίαση εικόνας, άλλα αφορά όλη την εφαρμογή (όπως για παράδειγμα η επιλογή εικόνας, κλάσης ταξινόμησης, σχήματος άλλα και στατιστικών δεδομένων). Τελευταία

αρχικοποίηση αντικειμένου είναι για την κλάση JobsToDO. Η κλάση αυτή λαμβάνει ως είσοδο ασύγχρονες διεργασίες για την αποθήκευση των δεδομένων του χρηστή και τις εκτελεί παράλληλα η σειριακά ανάλογά το είδος της διεργασίας.

Η συνάρτηση init της κλάσης Master.js πραγματοποιεί την εκτέλεση όλων των λειτουργιών των αντικείμενων που αρχικοποιήθηκαν. Ο τύπος της λειτουργίας των εσωτερικών αντικειμένων είναι ασύγχρονος και οπότε για να υπάρχει επιβεβαίωση ότι η αρχικοποίηση θα πραγματοποιηθεί ορθά, οι εκκίνηση των αντικειμένων (μέθοδος init) γίνεται εμφωλευμένα μέσω Callback Functions [22]. Δηλαδή μόλις τελειώσει η εκτέλεση των εσωτερικών αντικείμενων (μεθοδο init) τότε η συνάρτηση καλεί τον εαυτό της με σκοπό να εκτελέσει το επόμενο εμφωλευμένο init. Αρχικά στην κεντρική συνάρτηση init, δημιουργείται ένα ανώνυμο αντικείμενο με όνομα ReceiveAllEssentialsService που αναλαμβάνει το ασύγχρονη AJAX (Asynchronous JavaScript and XML) request στο σύστημα υποστήριξης της πλατφόρμας ώστε να αποθηκεύσει τα αντικείμενα των κλάσεων ταξινόμησης και σχήματα στην μνήμη της πλατφόρμας. Έπειτα πραγματοποιείται η εκτέλεση των μεταφράσεων και το αρχείο σύμφωνα με την προσβασιμότητα του χρηστή αποθηκεύεται στην κεντρική μνήμη της εφαρμογής ώστε να είναι προσπελάσιμη από όλα τα εμφωλευμένα αντικείμενα.

```
let getUrl = window.location.toString();
if (!("localDomain" in window)) {
    window.localDomain = getUrl.substr(0, getUrl.lastIndexOf("/"));
}
if (!("urlConnectionApi") in window) {
    window.urlConnectionApi = null;
}
let translation = new Translation();
class Master {
    constructor() {
        this.translation = translation;
        this.toast = toast;
        this.currentImageTab = null;
        this.user = null;
        this.userConfiguration = null;
        this.models = {
            shape_types: new Map(),
            classifications: new Map()
        };
        this.imageTabs = new Map();
        this.tabHandler = new TabHandlerUI();
        this.jobsToDo = new JobsToDo();
    }
    init(callback = () => {
    }) {
        let self = this;
        new ReceiveAllEssentialsService(() => {
            this.toast.init(() => {
                this.translation.init(() => {
                    this.tabHandler.init(() => {
                        let imageTabUI = this.tabHandler.selectImageTabUI;
                        if (this.userConfiguration && this.user &&
imageTabUI.images[this.userConfiguration.attribute_parser_image_id]) {
                            let image =
this.tabHandler.selectImageTabUI.images[
                            this.userConfiguration.attribute_parser_image_id
+ (this.userConfiguration.image_testing_id ? "|" +
this.userConfiguration.image_testing_id : null)];
                            let testing_appender = "[data-testing='']";
                            let image_id = image.id;
                            if (image.testing_id) {
                                testing_appender = "[data-testing='" +
image.testing_id + "']";
                                image_id += "|" + image.testing_id;

                            }

                            let $li = $(imageTabUI.childNode + " ul
li:not(.dropdown-header) img[data-id='" + image.id + "']" +
testing_appender).parents("li");
                            if ($li.length) {

                                imageTabUI.onChangeImage(image_id, null, $li,
null)
                                imageTabUI.setActiveClass($li);
                            }
```

Εικόνα 30: Πηγαίος κώδικάς του Master.js αρχείου.

4.2.2 ContextEvent.js

Το αρχείο ContextEvent.js είναι αναλαμβάνει την αναγνώριση των συμβάντων που εκτελούνται από τον χρηστή στον καμβά και κατευθύνει την λειτουργικότητα στο κατάλληλο αντικείμενό ή μέθοδο. Η λειτουργικότητα αυτή είναι καθιστά την διαχείριση και αλληλεπίδρασή του χρήστη με την εφαρμογή ομαλή και εύκολη.

Σχετικά με τον πηγαίο κώδικα της εφαρμογής (Εικόνα 31), η πιο σημαντική συνάρτηση είναι η _paintObject. Η συνάρτηση αυτή ενεργοποιείται όταν πραγματοποιείται ένα συμβάν κλικ (αριστερό click event από το ποντίκι) με σκοπό να δημιουργήσει ευρήματα πάνω στην εικόνα σύμφωνα με τις επιλογές που έχουν καθοριστεί. Αρχικά δηλώνεται και επεξεργάζεται η σχετική τιμή των συνταγμένων του ποντικού και έπειτα εάν το τρέχον σχήμα υπάρχει ήδη και είναι πολύγωνο τότε συνεχίζεται η σχεδίαση του. Στην συνέχεια, όταν ο χρήστης επιλεγεί ένα σχήμα, τότε το σχήμα γίνεται «τρέχον» και σχεδιάζεται ένας μετατροπέας εξωτερικά από το αντικείμενό για επεξεργαστεί ή να μετακινήσει στο καμβά. Ενώ εάν δεν συμβεί κανένα από αυτά τα γεγονότα τότε καλείτε η συνάρτηση mouse-up που θα αναλάβει να δημιουργήσει καινούριο σχήμα στις συγκεκριμένες συντεταγμένες του καμβά.

```
_paintObject(e) {
    let position = this.pointerPosition;
    let self = this;
    if (this.currentObject && this.currentObject.lockingMechanism &&
!this.currentObject.final_form) {
        return this.currentObject.movePosition(position)
    }
    let leftClick = e.evt.button === 0;
    if (!leftClick) return;
    let target = e.target;
    if (target._id && this.currentObjects[target._id]) {
        this.currentObject = this.currentObjects[target._id];
        if (!this.currentObject.shape.getDraggable())
            this.currentObject.shape.draggable(true);
    } else {
        this.currentObject = null;
    }
    if (this.isReadyToPaint(position)) {
        this.creatingObject = false;
        this.setCurrentObjectTransformer(e);
        return false;
    } else
        this.removeAllTransformers(e);
    this.painting = true;
    this.finalLayerRendered = false;
    this.creatingObject = true;
    this.mouseup(e);
}
```

Εικόνα 31: Πηγαίος κώδικάς συνάρτησής _paintObject, κλάση ContextEvent

4.2.2 KonvaShapeF.js

Η κλάση KonvaShapeF είναι υπεύθυνη για τη δημιουργία όλων των σχημάτων και είναι μια από τις κλάσεις που κληρονομούνται από όλα τα σχήματα της πλατφόρμας. Περιέχει όλα τα χαρακτηριστικά (μεταβλητές) που χρειάζονται τα αντικείμενα για να εμφανιστούν στον καμβά χρησιμοποιώντας την βιβλιοθήκη γραφικών KonvaJS.

Οι ιδιότητες των αντικείμενων(σχημάτων) που χρησιμοποιούμε από την βιβλιοθήκη KonvaJS βρίσκονται στην συνάρτηση config στον πηγαίο κώδικα της Εικόνας 32. Τα χαρακτηριστικά αυτά περιλαμβάνουν το χρώμα, το εξωτερικό περίγραμμα, τη σκίαση και άλλες πολλές γραφικές διαμορφώσεις για τα σχήματα. Ωστόσο, τα τελικά σχήματα χρησιμοποιούν και άλλα πιο περιπλοκά χαρακτηριστικά και ιδιότητες ανάλογά με τον τύπο του. Το παραγόμενο αποτέλεσμα ικανοποιεί πλήρως τις λειτουργικές απαιτήσεις του χρήστη και το περιβάλλον είναι φιλικό στη χρήση και στην αξιοποίηση από αυτόν.

```
config() {
    this.fill = this.fill ? this.fill : this.currentColor;
    this.stroke = this.stroke ? this.stroke : "black";
    this.shadowColor = this.shadowColor ? this.shadowColor : "black";
    this.konvaConfig = {
        x: this.x,
        y: this.y,
        width: this.defaultWidth === this.width ? this.defaultWidth :
this.width,
        height: this.defaultHeight === this.height ? this.defaultHeight :
this.height,
        radius: this.radius,
        scaleY: this.scaleY,
        scaleX: this.scaleX,
        radiusX: this.radiusX,
        radiusY: this.radiusY,
        points: this.points,
        draggable: this.draggable,
        fill: this.fill,
        listening: this.listening,
        opacity: this.opacity,
        stroke: this.stroke,
        ignoreStroke: this.ignoreStroke,
        strokeWidth: this.strokeWidth,
        strokeScaleEnabled: false,
        perfectDrawEnabled: this.perfectDrawEnabled,
        shadowForStrokeEnabled: false,
        shadowColor: this.shadowColor,
        shadowBlur: this.shadowBlur,
        shadowOffset: this.shadowOffset,
        shadowOpacity: this.shadowOpacity,

    };
    this.positionCreated = this.pos;
}
```

Εικόνα 32: Πηγαίος κώδικας μεθόδου config, κλάση KonvaShapeF.

4.2.4 CollectDataShapes.js

Η αρχείο CollectDataShapes έχει δημιουργηθεί για να συλλέγει τις νέες εισαγωγές και τροποποιήσεις μασκών σε κάθε καμβά ξεχωριστά, και για να τις εισάγει με αποστολή μέσω ασύγχρονης κλήσης στο σύστημα υποστήριξης. Το αντικείμενο αυτό παρέχει την δυνατότητα στον καμβά να συλλεγεί αυτόματα τις μάσκες που πρέπει να αποθηκευτούν στη βάση δεδομένων ακόμα και αν ο χρήστης εκτελεί τις ενέργειες με μεγάλη ταχύτητα.

Το πιο σημαντικό συστατικό της κλάσης αυτής είναι η μέθοδος checkJobAndPush, στην οποία τα δεδομένα περνούν στην κατάλληλη κλήση AJAX. Αρχικά, όταν συμβαίνει κάποια αλλαγή η κλάση αυτή ενεργοποιείται και τα δεδομένα συλλέγονται και κλωνοποιούνται για να διατηρηθεί η κατάσταση τους. Αυτό συμβαίνει λόγω της ασύγχρονης φύσης και δεδομένου ότι υπάρχει ενδεχόμενο να συμβεί κάποια αλλαγή στα αντικείμενα από τον χρηστή πριν το αποτέλεσμα της ανταπόκρισης AJAX. Επίσης μέσα στην ουρά αντικειμένων SaveShapesService δηλώνεται και η συνάρτηση Callback που θα εκτελεστεί μόλις η διαδικασία ολοκληρωθεί και εφόσον ολοκληρωθούν τα αντικείμενα (μάσκες) αυτά θα ενημερωθούν χωρίς ο τελικός χρήστης να κάνει κάποια άλλη ενέργεια.

```
checkJobAndPush() {
    let data = this.allData;
    let self = this;
    if (data.length && !this.job_inited) {
        master.jobsToDo.push(SaveShapesService, Object.assign({},
this.attributesParsed), (res) => {
            if (res) {
                this.job_inited = false;
                this.updateAllDataToSaved(res, data);
                if (self.tab.currentImage.model &&
self.tab.currentImage.model.name) {
                    //toast.showSuccess(translation.get("upload_shape"),
translation.get("shape_uploaded") + self.tab.currentImage.model.name);
                }
            }
        }, this.job_id);
        this.job_inited = true;
    } else if (!data.length && this.job_inited) {
        master.jobsToDo.popJobById(this.job_id);
        this.job_inited = false;
    }
}
```

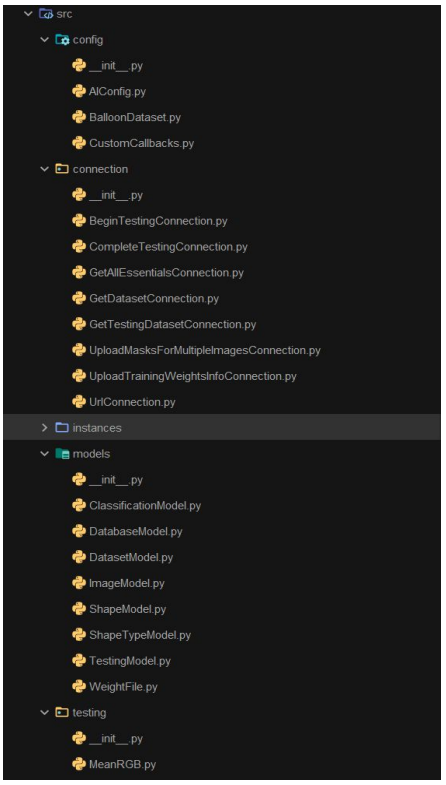
Εικόνα 33: Πηγαίος κώδικας της checkJobAndPush, κλάση CollectDataShapes

4.3 Υποσύστημα Τεχνητής Νοημοσύνης (Artificial Intelligence)

Το υποσύστημα τεχνητής νοημοσύνης (Artificial Intelligence app) είναι ο κορμός όλης της πλατφόρμας για την κατάτμηση των εικόνων. Αποτελεί το υποσύστημα για το οποίο η

πλατφόρμα δημιουργήθηκε, οπότε θα μπορούσε να αναφέρει κανείς ότι τα αλλά δυο υποσυστήματα είναι βοηθητικά και χρησιμοποιούνται για την παρουσίαση στο τελικό χρήστη. Το υποσύστημα τεχνητής νοημοσύνης ευθύνεται για την εκμάθηση εικόνων και μασκών και την εξαγωγή αυτομάτων αποτελεσμάτων από το νευρωνικού δίκτυο. Οι δυο κυρίες διαδικασίες έχουν εξολοκλήρου παραλληλοποιηθεί στους επεξεργαστές (CPUs) και στους επεξεργαστές των καρτών γραφικών (GPUs) του συστήματος, με αποτέλεσμα το σύστημα να είναι επεκτάσιμο και να μπορεί να καλύψει, τις οποίες ανάγκες προκύψουν σε επεξεργαστική ισχύ από τους χρήστες. Σύμφωνα με τις δυο διαδικασίες αυτές θα εξηγηθεί και η λειτουργικότητα του συστήματος.

Επιπλέον, οι κυρίες τεχνολογίες που χρησιμοποιήθηκαν για την υλοποίησή τους συστήματος είναι α) η υψηλών επιδόσεων αντικειμενοστραφής προγραμματιστική γλώσσα Python και β) η βιβλιοθήκη Mask-RCNN με ραχοκοκαλιά του γνωστό συνελικτικό δίκτυο ResNet. Ενώ, η αρχιτεκτονική του συστήματος είναι πλήρως αντικειμενοστραφής και έχει δημιουργηθεί στο πλαίσιο της λειτουργικότητας του Mask-RCNN. Ωστόσο χρησιμοποιήθηκαν και άλλες βοηθητικές βιβλιοθήκες και τεχνολογίες που εξυπηρετούν την λειτουργικότητα του συστήματος όπως η τεχνολογία CUDA, η οποία είναι ένα API για τον παραλληλισμό στις κάρτες γραφικών, και η ανοιχτού λογισμικού βιβλιοθήκη νευρωνικών δικτύων Keras που εκτελείται με βάση το TensorFlow. Σχετικά με τις κλάσεις ταξινόμησή των ευρημάτων των ιστολογικών εικόνων αυτές, η λιπώδης διήθηση (fat), τα διογκωμένα κύτταρα (ballooning), η φλωγμωνή (inflammation), τα σιγμοειδή (sinusoid), τα αγγεία (vein) αλλά και τέλος ολόκληρο το παρέγχυμα της ιστολογικής περιοχής (area).



Εικόνα 34: Δομή του πηγαίου κώδικα

4.3.1 Εκπαίδευση (Training)

Η εκπαίδευση νευρωνικού δικτύων είναι μια επίπονη χρονοβόρα διαδικασία που κρατάει μέρες υπολογίσιμου ακόμα και στα σύγχρονα συστήματά, και στοχεύει στην μεταλαμπάδευση της γνώσης στο δίκτυο ώστε να είναι ικανό να αναγνωρίζει και αντικείμενα και μάσκες τμηματοποίησης σε εικόνες. Στο κεφάλαιο αυτό αναλύουμε με λεπτομέρεια τα βήματα που ακολουθηθήκαν και τον πηγαίο κώδικα που χρησιμοποιήθηκε για την ολοκλήρωση της διαδικασίας. Ωστόσο, θα παρουσιαστεί επιγραμματικά η σειράς των βημάτων για να γίνει κατανοητή η διαδικασία από τον αναγνώστη.

Τα βήματα που εκτελούνται από το σύστημα για να ολοκληρωθεί η διαδικασίας εκπαίδευσης του Mask R-CNN και η εξαγωγής ενός αποδοτικού μοντέλου, αρχικά με την δημιουργία δυο συνόλων δεδομένων (εκπαίδευσης και επικύρωσης) με εικόνες και μάσκες από τους διαχειριστές του συστήματος υποστήριξης. Το συγκεκριμένο σύνολο που χρησιμοποιήθηκε αποτελείται από 80 εικόνες και 20000 μάσκες και διαχωρίστηκε 67% για την εκπαίδευση και 33% για την επικύρωσή. Έπειτα, οι εικόνες και οι συντεταγμένες για τις μάσκες αποθηκεύονται τοπικά μέσω API του συστήματος υποστήριξης και δημιουργούνται καινούρια στιγμιότυπα των αντικειμένων από το Mask R-CNN. Εν συνεχεία με σκοπό να

δημιουργήσουμε επιπλέον ποικιλομορφία στα ευρήματα εκπαίδευσης δημιουργείται ένα τυχαίο και δυναμικό επαυξημένο σύνολο δεδομένων (augmented) το οποίο προστίθεται στο αρχικό σύνολο. Τέλος δημιουργούνται συναρτήσεις Callback για την αποθηκευτή των στατιστικών και των αρχείων με τα βάρη του νευρωνικού (weight files), τα οποία εισάγονται στην εγγενή συνάρτηση εκπαίδευσης fit_generator του Keras, ως παράμετροι μαζί με το σύνολο δεδομένων και άλλες οδηγίες. Τα στατιστικά αποτελέσματα των επιδώσεις και οι σχετικές διαδρομές των τελικών βαρών αποστέλνονται μέσω API στον κεντρικό διακομιστή του συστήματος.

4.3.1.1 Δημιουργία του Dataset

Όπως αναφέρθηκε και ανωτέρω, το σύνολο δεδομένων διαχωρίζεται σε σύνολο εκπαίδευσης και επικύρωσης. Το σύνολο εκπαίδευσης χρησιμοποιείται για την εκπαίδευση του νευρωνικού δικτύου ενώ το σύνολο επικύρωσης για την εξαγωγής στατιστικών λαθών που περιγράφουν πρόοδος της διαδικασίας. Η σημαντικότερη κλάση της διαδικασίας είναι η κλάση LoadFromServer, η οποία αναλαμβάνει την δημιουργία στιγμιότυπων και διαχείριση της δομής των καταλογών για κάθε εικόνα του συνόλου, αλλά και δημιουργεί για κάθε σύνολο δεδομένων ένα στιγμιότυπο του ατνικειμένου SetInstance ξεχωριστά (Εικόνα 35). Η κλάση αυτή ευθύνεται για την δημιουργία στιγμιότυπων από τις εικόνες καθώς και τις πληροφορίες για αυτές, ενώ πραγματοποιεί και την τοπική αποθήκευση του σκέλος JSON που αντιστοιχεί το σύνολο αυτό. Για παράδειγμα η συνάρτηση download_images αναλαμβάνει να φέρει και να αποθηκεύσει τοπικά στην συσκευή το JSON αρχείο. Σε περίπτωση που η εικόνα υπάρχει ήδη στην κατάλληλη διαδρομή δεν γίνεται κάποια ενέργεια ενώ εάν ή εικόνα έχει διαφοροποιηθεί από το σύστημα υποστήριξης και δεν υπάρχει πια τότε θα διαγραφτεί.

```
class SetInstance:
    image_models = None
    set_json = []
    type = "validation",
    dataset_model = None
    current_path = ""
    image_names = None
    global_path = None
    def __init__(self, type_set: str, dataset_model, path_dataset: str,
set_json):
        self.set_json = set_json
        self.type = type_set
        self.path = path_dataset;
        self.global_path = path
        self.dataset_model = dataset_model
        self.current_path = path_dataset + "/" + str(self.type) + "/"
        self.image_names = []
        self.image_models = []
        self.image_names = [f for f in listdir(self.current_path) if
isfile(join(self.current_path, f))]
        self.init_images()
        self.download_images()
        self.save_json_set()
    def init_images(self):
        for index, key in enumerate(self.set_json):
            single_file = self.set_json[key]
            if "image_attributes" in single_file:
                current_image = ImageModel(**single_file["image_attributes"])
                self.image_models.append(current_image)

    def download_images(self):
        for model in self.image_models:

            if not path.exists(self.current_path + model.canonical_name):
                try:
                    file = open(self.current_path + model.canonical_name,
'wb')
                    print(model.image)
                    file.write(request.urlopen(model.image).read())
                    file.close()
                except Exception as e:
                    print(e)
            if model.canonical_name in self.image_names:
                self.image_names.remove(model.canonical_name)
        # delete other
\
        for name in self.image_names:
            if os.path.exists(self.current_path + name) and "set.json" not in
name:
                os.remove(self.current_path + name)

    def save_json_set(self):
        with open(self.current_path + "set.json", 'w') as f:
            json.dump(self.set_json, f, ensure_ascii=False)
```

Εικόνα 35: Πηγαίος κώδικας της Python κλάσης SetInstance.

4.3.1.2 Augmentation

Η διαδικασία της δημιουργίας του επαυξημένου συνόλου (augmentation) των δεδομένων χρησιμοποιείται για την αύξηση εικόνων και μασκών με καινούρια δεδομένα που δεν εχουν εισαχθεί από τους χρήστες άλλα βγαίνουν δυναμικά και τυχαία από την πλατφόρμα.

Προσφέρει στο νευρωνικό δίκτυο πιο αποτελεσματική εκπαιδεύση και λύνει το πρόβλημά της υπερεκπαίδευσης (overtrain).

Στην παρούσα διπλωματική εργασία η διαδικασία που χρησιμοποιήθηκε είναι σχετικά απλή και δεν αλλάζει πολύ την μορφή των εικόνων και μασκών. Ο κώδικας που χρησιμοποιείται για την διαδικασία της δημιουργίας του επαυξημένου συνόλου φαίνεται στην Εικόνα 36. Σύμφωνα με την προσέγγιση που χρησιμοποιήθηκε το σύνολο όλων των επαυξημένων δεδομένων δεν ξεπερνούν το 50% του συνολικού dataset, με σκοπό να μην αλλοιώνεται η φύση του του συνόλου, και να επιτευχθεί αποδοτικότερη έξοδο του νευρωνικού δικτύου για τις ιστολογικές εικόνες. Το ποσοστό αυτό είναι εμπειρικό και διαφέρει ανάλογά το πρόβλημα, τα δεδομένα του προβλήματος, και την φύση των αποτελεσμάτων.

Επίσης οι διαδικασίες/οδηγίες που ακολουθούνται για την αξιοποίηση των μασκών και εικόνων φαίνονται στον πηγαίο κώδικα της Εικόνας 36. Αρχικά γίνεται περιστροφή των εικόνων και μασκών είτε στις 90 μοίρες, και θόλωση κατά 5% ώστε να κάνουν πιο δύσκολο τον εντοπισμό από το νευρωνικό δίκτυο. Ως αποτέλεσμα δύναται να αναγνωρίσει ακόμα και πιο ακραίες περιπτώσεις εικόνων. Στην συνέχεια για το 20% των εικόνων παραμορφώνονται τα χρώματα στον χρωματικό χώρο RGB που βασίζεται η εφαρμογή. Τέλος γίνεται επιστροφή και κλιμάκωση των εικόνων ώστε το σύνολο που εφαρμόζουμε είναι μέρη των αρχικών εικόνων και έτσι ώστε να συγκρατηθούν οι απαιτήσεις για RAM και VRAM (μνήμη των καρτών γραφικών) του συστήματος στο ελάχιστο.

```
class Augmentation:
    @staticmethod
    def get_full():
        return imgaug.augmenters.Sometimes(0.5,
imgaug.augmenters.Sequential([
            imgaug.augmenters.OneOf([
                imgaug.augmenters.Fliplr(0.5),
                imgaug.augmenters.Flipud(0.5),
            ]),
            imgaug.augmenters.Sometimes(0.05,

imgaug.augmenters.GaussianBlur(sigma=(0, 0.5))
                                        ),
            imgaug.augmenters.Sometimes(0.2,
                                        imgaug.augmenters.Multiply((0.8,
1.2), per_channel=0.2)
                                        ),
            imgaug.augmenters.Affine(shear=(-8, 8)),
            imgaug.augmenters.OneOf([
                imgaug.augmenters.Affine(rotate=(-45, 45)),
                imgaug.augmenters.Affine(rotate=(-90, 90)),
            ]),
            imgaug.augmenters.OneOf([
                imgaug.augmenters.Affine(scale=(0.5, 0.8)),
                imgaug.augmenters.Affine(scale=(1 / 1.2, 1 / 1.5)),
                imgaug.augmenters.Affine(scale=(1.8, 2.1)),
                imgaug.augmenters.Affine(scale=(2.4, 2.7)),
                imgaug.augmenters.Affine(scale=(1 / 3, 1 / 3.3)),
                imgaug.augmenters.Affine(scale=(1 / 3.6, 1 / 3.9)),

           ])

        ], random_order=True))
```

Εικόνα 36: Πηγαίος κώδικας του Augmentation

4.3.1.3 Διαμόρφωση Mask R-CNN

Για τις ανάγκες της πλατφόρμας οι διαμορφώσεις που δοκιμάστηκαν του αρχικού Mask R-CNN ήτανε πολλές με σκοπό να επιτευχθεί το επιθυμητό αποτέλεσμα σύμφωνα με την κρίση και τις απαιτήσεις του χρήστη. Οι πιο σημαντικές διαμορφώσεις βρίσκονται στον πηγαίο κώδικα της κλάσης AIConfig. Στην παρούσα κλάση που κληρονομήθηκε από την κλάση Config, θα αναλύσουμε τις πιο σημαντικές αλλαγές των μεταβλητών για τις ανάγκες του έργου. Αρχικά είναι η υπερπαραμέτρος learning_rate με τιμή 0.002. η παράμετρος αυτή αποφασίζει το ποσοστό κατά το οποίο θα τροποποιηθεί το μοντέλο και για να υπολογιστεί εκτιμώμενο σφάλμα που κάθε φορά ενημερώνει τα βάρη του μοντέλου. Για τον υπολογισμό της μεταβλητής MEAN_PIXEL δημιουργήθηκε διαφορετική εφαρμογή που μετράει την μέση τιμή των τιμών των χρωμάτων για όλες τις εικόνες του κεντρικού συστήματος. Μια πολύ σημαντική παράμετρος είναι το RPN_ANCHOR_SCALES που καθοδηγεί το μέγεθος των πλαισίων του δικτύου Region Proposal Network (RPN). Οι τιμές που επιλεχθήκανε είναι

σχετικά πολύ μικρές με σκοπό το μέγεθος τον αντικειμένων που επιχειρείται να εντοπιστεί ενδέχεται να είναι έως και το 0.1% της εικόνα, δηλαδή εξαιρετικά μικρό. Στην συνεχεία για να πραγματοποιηθεί η εκπαίδευση του νευρωνικού δικτύου απαιτούνται όλες οι εικόνες να έχουν τις ιδίες διαστάσεις. Αυτό όμως είναι αδύνατο σύμφωνα με τις μη λειτουργικές απαιτήσεις του χρήστη, οπότε εφαρμόστηκε περικοπή (crop) ως η τεχνική αλλαγής μεγέθους. Στη τεχνική αυτή γίνεται τυχαίες περικοπές μερών της εικόνας με αποτέλεσμα την αλλαγή μεγέθους της τελικής εικόνας ώστε να φτάσει την τιμή 512 του IMAGE_MIN_DIM. Να σημειωθεί πως η παραμόρφωση της εικόνας γίνεται σε συνδυασμό με τις μάσκες ώστε να μην γίνει λανθασμένα η συγγραφή των συνταγμένων των μασκών.

```
NAME = "x-balloon"
IMAGES_PER_GPU = 1
LEARNING_RATE = 0.002
GPU_COUNT = 2
POST_NMS_ROIS_TRAINING = 10000
POST_NMS_ROIS_INFERENCE = 10000
# Number of training steps per epoch
MAX_EPOCHS = 300
STEPS_PER_EPOCH = 500
VALIDATION_STEPS = 50
IMAGE_MIN_DIM = 512
IMAGE_MAX_DIM = 512
IMAGE_MIN_SCALE = 1
TRAIN_ROIS_PER_IMAGE = 1024
MAX_GT_INSTANCES = 1024
RPN_TRAIN_ANCHORS_PER_IMAGE = 1024  # 300
# Max number of final detections per image
DETECTION_MAX_INSTANCES = 1024
# 50 hidden layers
BACKBONE = "resnet101"
IMAGE_RESIZE_MODE = "crop"
MEAN_PIXEL = np.array([226.78973934096723, 207.18995435393035,
232.21716953829852])
RPN_ANCHOR_SCALES = (8, 16, 32, 64, 128)
DETECTION_MIN_CONFIDENCE = 0.5
```

Εικόνα 37: Μερικός πηγαίος κωδικός της κλάσης AIConfig

4.3.1.4 Διαδικασία εκπαίδευσης

Η διαδικασίας της εκπαίδευσης είναι το τελικό βήμα για την παραγωγή της αρχιτεκτονικής και των αρχείων με τα βάρη του τελικού νευρωνικού δικτύου. Ενώ η απόδοση του αξιολογείται σύμφωνα με τις εκτιμώμενες απώλειες που παράγονται από το Keras και το Mask R- και θα αναλυθούν με λεπτομέρεια σε συνδυασμό με τον πηγαίο κώδικα της κεντρική διαδικασία εκπαίδευση του νευρωνικού δικτύου στο κεφάλαιο αυτό.

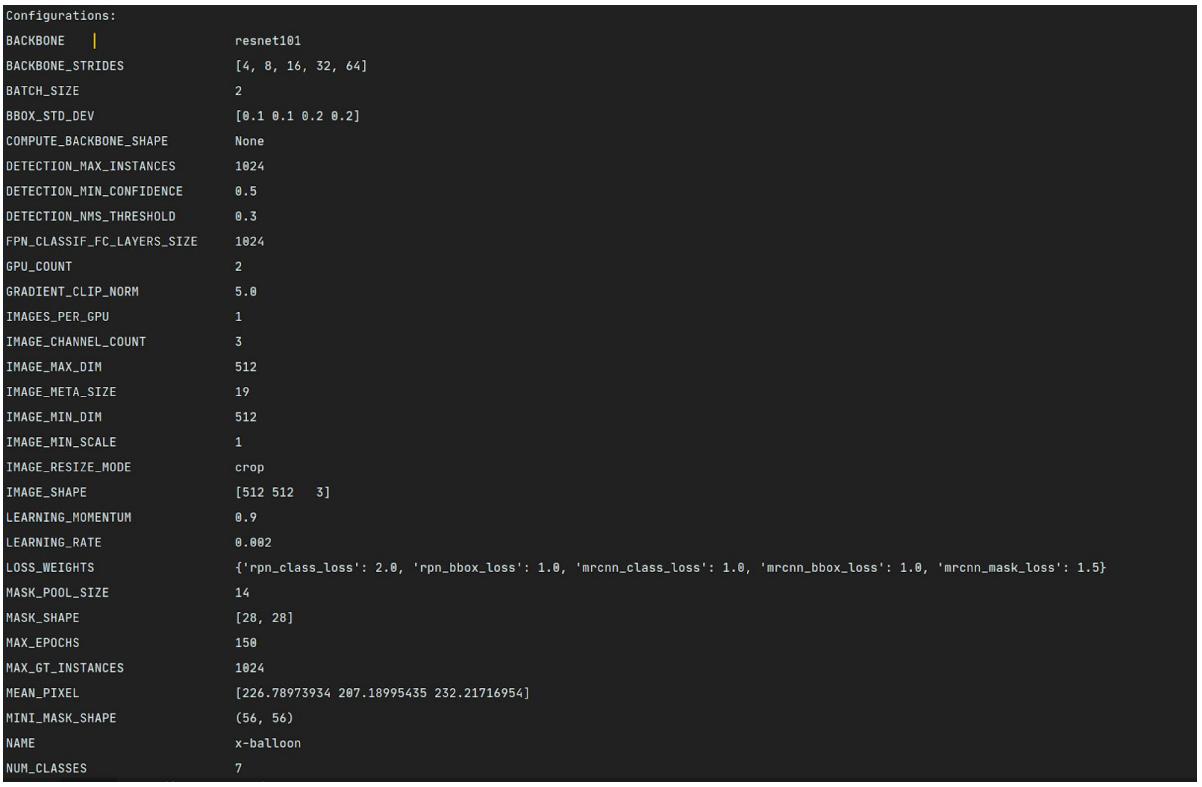
Η συνάρτηση για την διαδικασία εκπαίδευσης του νευρωνικού δικτύου (Mask R-CNN)

δέχεται ως παραμέτρους, το σύνολο εκπαίδευσης και το σύνολο επικύρωσής, που είναι απαραίτητα για οποιοδήποτε πρόβλημά και εφαρμογή. Εκτός των συνόλων δέχεται την παράμετρο του αριθμού των εποχών (epochs), οπού μια εποχή είναι το άθροισμα τον επαναλήψεων της εκπαίδευσης (training) και της επικύρωσης (validation) των εικόνων και μασκών. Στην συνέχεια δέχεται άλλες μεταβλητές όπως την augmentation, learning_rate και πίνακα callback functions που αναφέρθηκαν και παραπάνω. Τέλος εξίσου σημαντική παράμετρος είναι η παράμετρος layers που αφορά τα επίπεδα (layers) του δικτύου που θα εκπαιδευτούν. Ενδεικτικά αναφέρουμε ότι η εφαρμογή χρησιμοποιεί ως αρχικό σύνολο βαρών το Coco weight file. Η επανεκπαίδευση ισχύει για το Region Proposal Network (RPN), για τον ταξινόμητη και τις μάσκες του δικτύου. δηλαδή η εκπαίδευση γίνεται σύμφωνα με τις ανάγκες του προβλήματος αυτού και δεν επεκτείνει σε κάποιο άλλο.

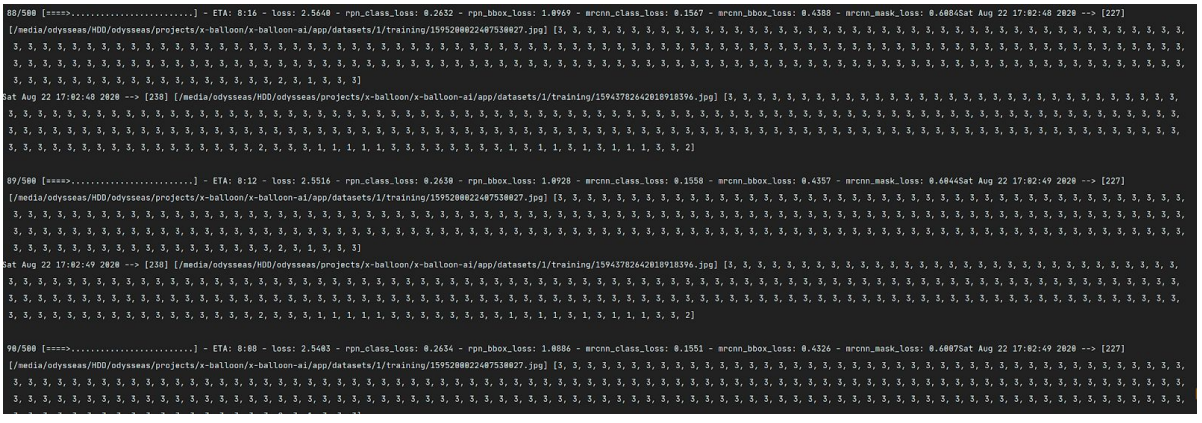
Επιπλέον, κατά την εκπαίδευση του νευρωνικού δικτύου προβάλλεται η εκτιμωμένη απώλεια (loss) ώστε να γίνεται κατανοητή η πρόοδος του Mask R-CNN μοντέλου στην πορεία των επαναλήψεων. Η εκτιμωμένη απώλεια είναι το άθροισμα των απωλειών rpn_class_loss , rpn_bbox_loss, mrcnn_class_loss, mrcnn_bbox_loss και mrcnn_mask_loss. Όπου rpn_class_loss είναι απώλεια του RPN ταξινομητή σε σύγκριση με τον rpn_box_class που αναφέρετε για το RPN γράφημα απώλειας πλαισίου οριοθέτησης. Ενώ οι τρεις υπόλοιπες loss τιμές αφορούν άμεσα το Mask-RCNN, δηλαδή το mrcnn_class_loss αφορά την απώλεια του ταξινομητή, το mrcnn_bbox_loss αφορά τη απώλεια για βελτίωση των περιγραμμάτων οριοθέτησης της μάσκας και το mrcnn_mask_loss αφορά την μάσκα δυαδικής απώλειας cross-entropy. Συμπεραίνοντας όσο πιο μικρές είναι αυτές οι απώλειες τόσο μικρότερη θα είναι η εκτιμώμενη απώλεια, οπότε το μοντέλο αποδίδει καλυτέρα αποτελέσματα.

```
self.model.train(self.dataset_train, self.dataset_val,
                 learning_rate=self.config.LEARNING_RATE,
                 # callbacks
                 custom_callbacks=[self.checkPointCallback,
self.metricsCallback],
                 epochs=self.config.MAX_EPOCHS,
                 layers=self.layers, augmentation=Augmentation.get_full())
```

Εικόνα 38: Συνάρτηση εκπαίδευσης του Mask R-CNN



Εικόνα 39: Configuration output από τον διερμηνευτή της Python



Εικόνα 40: Starting Output από την διαδικασία εκπαίδευσης του νευρωνικού δικτύου μέσω GPU

4.3.2 Εξαγωγή αποτελεσμάτων (Inference)

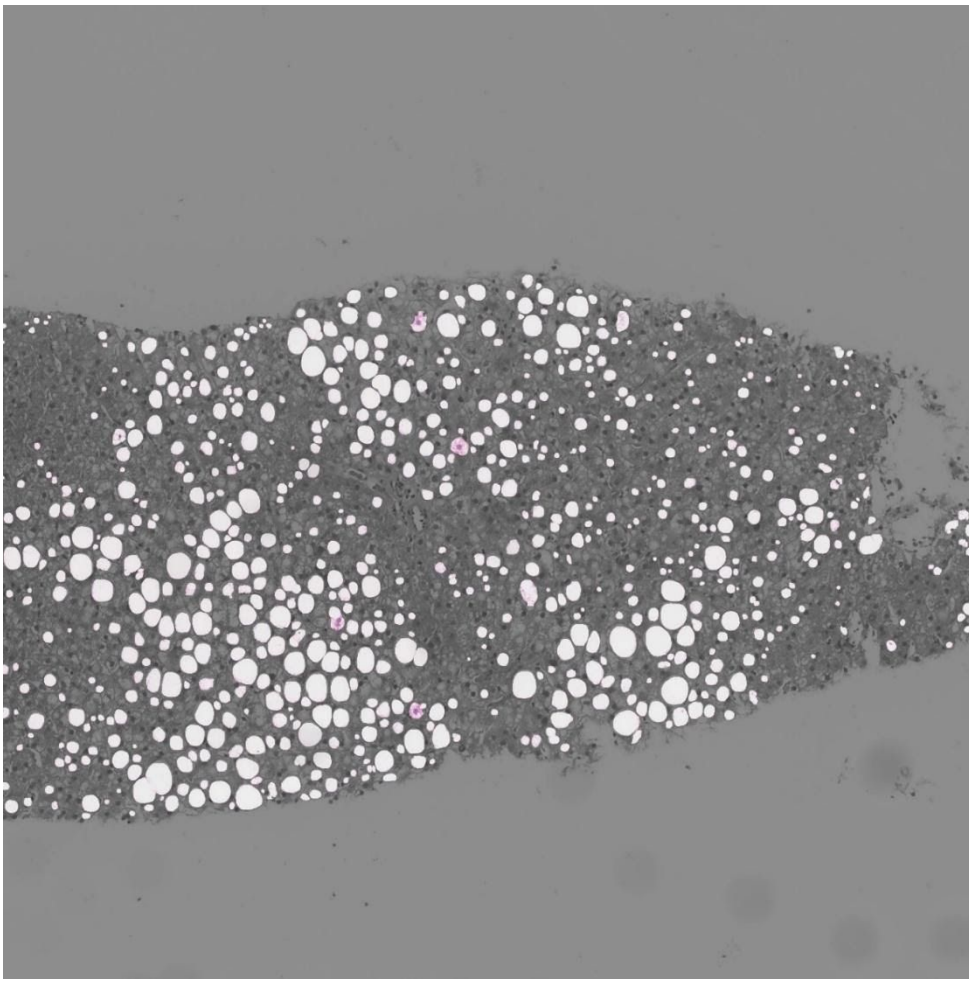
Η διαδικασία ελέγχου (testing procedure) έχει πολύ μικρό υπολογιστικό κόστος συγκριτικά με την εκπαίδευσης του νευρωνικού δικτύου αφού το απαιτούμενο μοντέλο δημιουργείτε και βελτιστοποιείται στο στάδιο της εκπαίδευσης. Στο κεφαλαίο αυτό θα γίνουν δοκιμές από τις εικόνες του συστήματος υποστήριξης και θα σχολιαστούν τα αποτελέσματα αλλά και ο πηγαίος κώδικας για τη διαδικασία αυτή.

Η διαδικασία που ακολουθείται από την πλατφόρμα για την φάση ελέγχου του Mask R-CNN μοντέλου έχει ως εξής: Αρχικά οι χρήστες δημιουργούν στο σύστημα υποστήριξης ένα σύνολο ελέγχου (testing set) σε συνδυασμό με τις ιστολογικές εικόνες που επιθυμούν να γίνει η αυτόματος εντοπισμούς των αντικείμενων (μάσκες). Στην συνέχεια η πλατφόρμα δέχεται το προ-εκπαιδευμένο μοντέλο (pre-trained model) και τις εικόνες του συνόλου ελέγχου, και για κάθε μια εκτελεί την σημασιολογική κατάτμηση. Το αποτέλεσμα του εντοπισμού είναι πολλές εικόνες σε δυαδική μορφή, οι οποίες έχουν τις διαστάσεις τις αρχικής εικόνας, ενώ στο σύνολο τους είναι όσες ακριβώς και οι μάσκες που έχει εντοπίσει το σύστημα σύμφωνα με την διαμόρφωση που έχει πραγματοποιθείς νωρίτερα στο Mask R-CNN. Ωστόσο αναφέρεται το Mask R-CNN επιστρέφει επιπλέον τις κλάσεις ταξινόμησης, τα περιγράμματα οριοθέτησης, τις περιοχές ενδιαφέροντος αλλά και τις ποσοστιαίες βεβαιότητες (confidence) για τις αντιστοιχίες δυαδικές εικόνες. Επειδή όπως αναφέρθηκε για το αποτέλεσμα των μασκών είναι δυαδικές εικόνες, γίνεται επεξεργασία για να βρεθούν τα βέλτιστα πολύγωνα των μασκών. Αφού ολοκληρωθεί η επεξεργασία των εικόνων και βρεθούν όλα τα πολύγωνα στις εικόνες, αυτά αποθηκεύονται στα καταλληλά αντικειμενοστραφή μοντέλα και αποστέλνονται στον κεντρικό διακομιστή ώστε να δοθεί η δυνατότητα στον τελικό χρηστή να δει το αποτέλεσμα μέσω της Επισημειωτή Εικόνων πλατφόρμας . Η παραπάνω διαδικασία ακολουθείται επαναληπτικά για όλες τις εικόνες και εκτελείται από τον πηγαίο κώδικα της εικόνας 41.

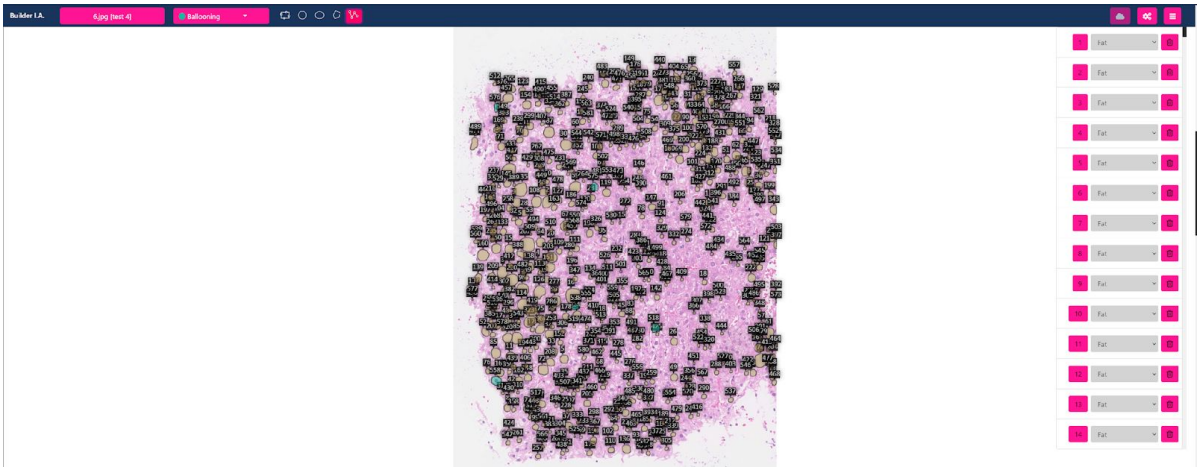
```
for index_images in range(len(images)):

    results = self.model.detect([images[index_images]], verbose=1)

    for index_result in range(len(results)):
        result = results[index_result]
        masks = result["masks"]
        scores = result["scores"]
        class_ids = result["class_ids"]
        image_model = testing_set.image_models[index_images]
        collector = GatherAllPolygons([image_model],
self.load_from_server.testingModel.id)
        splash = color_splash(images[index_images], masks)
        print(masks.shape[2], "masks")
        polygons = []
        classifications_ids = []
        score_lists = []
        for c in range(masks.shape[2]):
            polygon = Mask(masks[:, :, c]).polygons()
            if len(polygon.segmentation):
                # if one one mask there are multiple shapes
                for index_polygons in range(len(polygon.segmentation)):
                    polygons.append(polygon.segmentation[index_polygons])
                    classifications_ids.append(str(class_ids[c]))
                    score_lists.append(str(scores[c]))
        # print(score_lists)

        collector.push_polygon(image_model, polygons, classifications_ids,
score_lists)
        length = len(image_model.canonical_name)
        new_name = image_model.canonical_name[0:length - 4] + "-classified" +
image_model.canonical_name[

length - 4:length]
        # save image
        skimage.io.imsave(testing_set.current_path + new_name, splash)
        collector.upload()
CompleteTestingConnection(self.load_from_server.testingModel.id)
```

Εικόνα 41: Πηγαίος κώδικας που εκτελεί το testing στο Mask R-CNN μοντέλο για όλες τις εικόνες του testing set και ανεβάζει τις συνταγμένες των μασκών και τα στατιστικά στον κεντρικό διακομιστή

Παρακάτω φαίνονται παραδείγματα από την διαδικασία ελέγχου νέων ιστολογικών εικόνων. Για παράδειγμα στην ιστολογική εικόνα 42, βρέθηκαν 734 fat ευρήματα που εχουν άσπρο χρώμα και 7 balloon που σημαίνονται με μωβ χρώμα, με ποσοστιαία βεβαιότητα 97%.



Εικόνα 42: Ιστολογική εικόνα που έχει βγει από το αποτέλεσμα του Mask R-CNN testing



Εικόνα 43: Εμφάνιση στο Image Annotator με την αυτόματη εξαγωγή μασκών μέσων του Mask R-CNN testing

Κεφάλαιο 4

4. Συμπεράσματα και μελλοντικές επεκτάσεις

Στο τελευταίο κεφάλαιο με το οποίο κλείνει η παρούσα διπλωματική, πέρα από τον επίλογο της εργασίας, θα αναφερθούν συμπεράσματα και μελλοντικές επεκτάσεις της εφαρμογής.

4.1 Συμπεράσματα

Αρχικός στόχος της διπλωματικής εργασίας ήταν η ανάπτυξη εννοίας πλατφόρμα υπολογιστικού νέφους με σκοπό την αναγνώριση και σήμανση ιστολογικών εικόνων. Μέσα σε αυτά τα πλαίσια λόγο του όγκου, της πολυπλοκότητας και των μη λειτουργικών απαιτήσεων του χρήστη αποφασίστηκε το σύστημα να διασπαστεί σε τρία υποσυστήματα όπως αναλύθηκε με λεπτομέρεια. Το γεγονός αυτό είχε ως αποτέλεσμα την μεγάλη έκταση του πηγαίου κώδικα καθώς δημιουργήθηκε και η ανάγκη για προγραμματιστικών διεπαφών που επικοινωνούν οι πλατφόρμες μεταξύ τους.

Σχετικά με την αυτόματη αναγνώριση και σήμανση των ιστολογικών εικόνων, τα αποτελέσματα που εξάγονται από το νευρωνικό δίκτυο είναι αποδεκτά και ακριβή. επίσης δίνεται η δυνατότητα στον διαχειριστή του συστήματος να κληρονομεί και επανεκπαιδεύει τα βάρη των προ-εκπαιδευμένων μοντέλων αλλά. Η ιδιότητα αυτή του συστήματος συμβάλει ώστε όσο πιο πολύ τελικοί χρηστές διορθώνουν τα αποτελέσματα του νευρωνικού δίκτυού, τόσο περισσότερη γνώση αποκτά το σύστημα, και τα αποτελέσματα των μασκών θα γίνονται πιο ακριβή ακόμα σε ακρίβεια εικονοστοιχείο.

4.2 Μελλοντικές επεκτάσεις

Η πλατφόρμα υπολογιστικού νέφους που αναπτύχθηκε στα πλαίσια της διπλωματικής εργασίας είναι πλήρης ολοκληρωμένη και συμπαγής, άλλα απαιτεί την ενέργεια του διαχειριστή του συστήματος για να εκπαιδεύσει και να εξάγει αποτελέσματα από το

νευρωνικό δίκτυο.

Μια μελλοντική επέκταση η οποία θα μπορούσε να πραγματοποιηθεί είναι η προσθήκη ενός τέταρτου υποσυστήματος, το οποίο θα αναλάβει να επικοινωνεί τα γεγονότα των χρηστών μέσω διαδικτυακών υποδοχέων (web sockets). Με συνέπεια όλα τα υπόλοιπα υποσυστήματα της εφαρμογής να είναι συνδεμένα στον υποδοχέα αυτό και να δίδεται η δυνατότητα στους τελικούς πελάτες να εκπαιδεύουν και να εξάγουν αποτελέσματα από το υποσύστημα τεχνητής νοημοσύνης σε οποιαδήποτε χρονική στιγμή το επιθυμούν.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] A. Syromiatnikov, «A Journey Through the Land of,» 22 July 2020. [Ηλεκτρονικό]. Available: researchgate.net/profile/Danny_Weyns/publication/269303611_A_Journey_through_the_Land_of_Model-View-Design_Patterns/links/5e456578a6fdccd965a1ece8/A-Journey-through-the-Land-of-Model-View-Design-Patterns.pdf?origin=publication_list.

[2] A. M. a. I. Rau, «1,» [Ηλεκτρονικό]. Available: https://crimsonpublishers.com/prsp/pdf/PRSP.000505.pdf. [Πρόσβαση 22 7 2020].

[3] «1,» [Ηλεκτρονικό]. Available: https://en.wikipedia.org/wiki/Yii. [Πρόσβαση 10 6 2020].

[4] «WSDL Tutorial: Web Services Description Language with Example,» [Ηλεκτρονικό]. Available: https://www.guru99.com/wsdl-web-services-description-language.html. [Πρόσβαση 10 6 2020].

[5] R. Girshick, «Fast R-RCNN,» [Ηλεκτρονικό]. Available: https://arxiv.org/abs/1504.08083. [Πρόσβαση 10 6 2020].

[6] K. H. R. G. J. S. Shaoqing Ren, «Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks,» [Ηλεκτρονικό]. Available: https://arxiv.org/abs/1506.01497. [Πρόσβαση 10 6 2020].

[7] G. G. P. D. R. G. Kaiming He, «Mask R-CNN,» [Ηλεκτρονικό]. Available: https://arxiv.org/abs/1703.06870. [Πρόσβαση 10 5 2020].

[8] Firiuza, «ROI pooling vs. ROI align,» [Ηλεκτρονικό]. Available: https://medium.com/@Firiuza/roi-pooling-vs-roi-align-65293ab741db. [Πρόσβαση 11 6 2020].

[9] P. D. R. G. K. H. B. H. S. B. Tsung-Yi Lin, «Feature Network Pyramid for Object Detection,» [Ηλεκτρονικό]. Available: https://arxiv.org/abs/1612.03144. [Πρόσβαση 11 10 2020].

ΒΙΒΛΙΟΓΡΑΦΙΑ

[10] Python, «Garbage Collector interface,» [Ηλεκτρονικό]. Available: https://docs.python.org/3/library/gc.html. [Πρόσβαση 11 3 2020].

[11] «PHP,» [Ηλεκτρονικό]. Available: https://en.wikipedia.org/wiki/PHP. [Πρόσβαση 10 6 2020].

[12] Google, «V8 Runtime Overview,» [Ηλεκτρονικό]. Available: https://developers.google.com/apps-script/guides/v8-runtime. [Πρόσβαση 10 6 2020].

[13] «Javascript,» [Ηλεκτρονικό]. Available: https://en.wikipedia.org/wiki/JavaScript. [Πρόσβαση 10 6 2020].

[14] «MySQL,» [Ηλεκτρονικό]. Available: https://en.wikipedia.org/wiki/MySQL. [Πρόσβαση 11 6 2020].

[15] JavaTPoint, «MySQL Features,» [Ηλεκτρονικό]. Available: https://www.javatpoint.com/mysql-features. [Πρόσβαση 11 6 2020].

[16] HP, «The Evolutionary Development Model for,» [Ηλεκτρονικό]. Available: https://www.hpl.hp.com/hpjournal/96aug/aug96a4.pdf. [Πρόσβαση 12 6 2020].

[17] Nvidia, «Nvidia,» [Ηλεκτρονικό]. Available: https://www.nvidia.com/. [Πρόσβαση 14 10 2020].

[18] Nvidia, «Recommended GPU for Developers,» [Ηλεκτρονικό]. Available: https://developer.nvidia.com/cuda-gpus. [Πρόσβαση 12 6 2020].

[19] Bootstrap, «Bootstrap,» [Ηλεκτρονικό]. Available: https://getbootstrap.com/. [Πρόσβαση 13 7 2020].

[20] Morris, «Morris,» [Ηλεκτρονικό]. Available: https://morrisjs.github.io/morris.js/. [Πρόσβαση 11 7 2020].

[21] JQuery, «JQuery,» [Ηλεκτρονικό]. Available: https://jquery.com/. [Πρόσβαση 11 7 2020].

ΒΙΒΛΙΟΓΡΑΦΙΑ

[22] MDN , mozilla, «Callback Function,» [Ηλεκτρονικό]. Available: https://developer.mozilla.org/en-US/docs/Glossary/Callback_function. [Πρόσβαση 11 7 2020].