Ατομική Διπλωματική Εργασία

# ΑΥΤΟΜΑΤΟΠΟΙΗΜΕΝΟΙ ΕΛΕΓΧΟΙ ΓΙΑ ΟΠΤΙΚΑ ΣΦΑΛΜΑΤΑ ΣΥΣΤΗΜΑΤΩΝ ΔΙΑΔΙΚΤΥΟΥ ΜΕ ΕΝΣΩΜΑΤΩΣΗ ΤΟΥ Chat-GPT 4o ΚΑΙ ΟΔΗΓΟΥ ΦΥΛΛΟΜΕΤΡΗΤΗ PLAYWRIGHT ΣΕ C#

**Ιάσων Γεωργίου**

**ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ**

**ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ**

**Μάϊος 2025**


# Πανεπιστήμιο Κύπρου

# Τμήμα Πληροφορικής


## ΑΥΤΟΜΑΤΟΠΟΙΗΜΕΝΟΙ ΕΛΕΓΧΟΙ ΓΙΑ ΟΠΤΙΚΑ ΣΦΑΛΜΑΤΑ ΣΥΣΤΗΜΑΤΩΝ ΔΙΑΔΙΚΤΥΟΥ ΜΕ ΕΝΣΩΜΑΤΩΣΗ ΤΟΥ Chat-GPT 4o ΚΑΙ ΟΔΗΓΟΥ ΦΥΛΛΟΜΕΤΡΗΤΗ PLAYWRIGHT ΣΕ C#


**Ιάσων Γεωργίου**

Επιβλέπων

Δρ. Κωνσταντίνος Παττίχης


Αυτή η διπλωματική εργασία παραδόθηκε για την εκπλήρωση των υποχρεώσεων για την απονομή του πτυχίου Bachelor στην Επιστήμη Πληροφορικής στο Πανεπιστήμιο Κύπρου

Μάιος 2025

## Ευχαριστίες

Θα ήθελα να ευχαριστήσω τον Δρ. Κωνσταντίνο Παττίχη για την προσεκτική ακαδημαϊκή επίβλεψή του κατά την διάρκεια της εκπόνησης της διπλωματικής αυτής, και για τις χρήσιμες και ωφέλιμες εισηγήσεις του.

Θα ήθελα επίσης να ευχαριστήσω τον διευθυντή μου κ. Νίκο Αναστασίου που μου έδωσε την ευκαιρία και τους πόρους να αναπτύξω μια τόσο ενδιαφέρουσα εφαρμογή.

Ακόμα θα ήθελα να ευχαριστήσω και την σύζυγό μου, Οξάνα, για την υπομονή της και την στήριξή της καθόλη την διάρκεια των σπουδών μου.

## Περίληψη

Το πρόβλημα του αυτοματοποιημένου οπτικού ελέγχου και εύρεσης οπτικών σφαλμάτων είναι από τα λίγα προβλήματα στον κόσμο του αυτοματοποιημένου ελέγχου που παραμένουν ανοικτά. Η συνεισφορά των αυτοματοποιημένων ελέγχων στον χώρο ανάπτυξης λογισμικού και στον κύκλο ζωής του είναι αδιαμφισβήτητη και εάν επιλυόταν αυτό το πρόβλημα θα είχαμε μια εξαιρετικά σημαντική εξέλιξη στο χώρο του Quality Assurance και των αυτοματοποιημένων δοκιμών. Τα μοντέλα LLM και τα μοντέλα γενικής χρήσεως είναι ένας πιθανός τρόπος για την αντιμετώπιση αυτού του προβλήματος, καθώς είναι πολλαπλών μέσων και συνδυάζουν κείμενο και εικόνα που είναι και τα δύο απαραίτητα για την κατανόηση των περιεχομένων ενός συστήματος διαδικτύου, σε αντίθεση με μοντέλα επεξεργασίας εικόνας που παραμένουν σε καθαρά οπτικές αλλά όχι νοηματικές διαφορές. Σε αυτήν την διπλωματική, εξερευνάται ένα από αυτά τα μοντέλα, το Chat-GPT 4o, αφού αναπτύχθηκε και σχετική εφαρμογή διαχείρισης δοκιμών. Παρότι προς το παρόν δεν μπορεί να αυτοματοποιηθεί πλήρως αυτή η δουλεία, τα αποτελέσματα έδειξαν πως το συγκεκριμένο μοντέλο είναι πολλά υποσχόμενο και ελπιδοφόρο. Η αρχιτεκτονική της λύσης (Playwright, c#), προσφέρει σημαντικά περιθώρια επέκτασης, συμπεριλαμβανομένων δυνατοτήτων για παραμετροποιημένες δοκιμές, καλύτερη ερμηνεία αποκλίσεων, καθώς και ενσωμάτωση με συστήματα συνεχούς ολοκλήρωσης και ανάπτυξης (CI/CD). Η συγκεκριμένη προσέγγιση ανοίγει τον δρόμο για την ανάπτυξη ευφυών εργαλείων που είναι οικονομικώς βιώσιμα οπτικού ελέγχου, τα οποία μπορούν να ενισχύσουν την διαδικασία ποιοτικής διασφάλισης σε περιβάλλοντα με υψηλές απαιτήσεις πολυπλοκότητας, μεταβλητότητας και όγκου εργασίας.

**Περιεχόμενα**

# Κεφάλαιο 1

## Κεφάλαιο 1: Εισαγωγή



### 1.1 Οπτικός Έλεγχος

Στο χώρο της ανάπτυξης λογισμικού υπάρχουν πολλά διαφορετικά μοντέλα προσέγγισης του κύκλου ανάπτυξης. Σε όλα αυτά τα μοντέλα **[1]**, υπάρχει τουλάχιστον μια φάση δοκιμής (π.χ. Waterfall) ή πολλαπλές και επαναλαμβανόμενες (π.χ. agile). Σε κάποιες περιπτώσεις, είναι εφικτό να προχωρούν οι υπόλοιπες φάσεις ανάπτυξης λογισμικού, ως ένα βαθμό, παράλληλα με τους ελέγχους και άρα υπάρχει περίπτωση να μην αποτελεί κρίσιμο σημείο η φάση ελέγχου. Όμως, πολύ συχνά, ιδίως σε Agile μεθοδολογίες, προκύπτουν σημεία συμφόρησης όσον αφορά τις φάσεις δοκιμών **[2]**. Συνεπώς, ούτως ώστε να μειωθεί ο χρόνος αυτός των δοκιμών, πολλοί οργανισμοί ανάπτυξης λογισμικού προσπαθούν να ενσωματώσουν αυτοματοποιημένες δοκιμές στην φάση ελέγχου. Κυρίως, οι αυτοματοποιημένοι έλεγχοι αφορούν στη βασική λειτουργικότητα της διεπαφής χρήστη (με την χρήση Οδηγών Φυλλομετρητή – XPath), στην επεξεργασία των δεδομένων από την ιστοσελίδα, δοκιμές API, δοκιμές μονάδων κ.ο.κ. προσφέροντας μεγαλύτερη επανάχρηση δοκιμών, μεγαλύτερη επαναληπτικότητα, μεγαλύτερη κάλυψη δοκιμών και μείωση συνολικών ωρών εργασίας **[3]**, όμως είναι γνωστό τοις άπασι πως δεν επαρκεί ο αυτοματισμός από μόνος του, καθώς υπάρχουν τομείς που είναι σχεδόν αδύνατο να αυτοματοποιηθούν με ικανά υψηλή ακρίβεια.

Ένας από αυτούς τους τομείς, αφορά στον οπτικό έλεγχο (visual testing). Ο οπτικός έλεγχος, είναι το είδος ελέγχου λογισμικού που επικεντρώνεται στο να επικυρώνει ότι η

γραφική διεπαφή χρήστη (Graphical User Interface – GUI) *εμφανίζεται* και *λειτουργεί* καθώς αναμένεται. Συμπεριλαμβάνει την επικύρωση των οπτικών στοιχείων, όπως την διάταξη, τα χρώματα, τις γραμματοσειρές, τις εικόνες και τον τρόπο ανταπόκρισης των στοιχείων μεταξύ διαφόρων συσκευών, μεγεθών οθονών και αναλύσεων. Ως προς τις χειροκίνητες δοκιμές, η εμπειρία του ελεγκτή είναι αρκετή για να συγκρίνει με τα σχέδια της ιστοσελίδας. Όμως, ως προς τις αυτοματοποιημένες, δεν υπάρχουν ακόμα εργαλεία ικανά να προσφέρουν αυτονομία των αυτοματοποιημένων δοκιμών **[4]**, ενώ ταυτόχρονα, όπως προαναφέρθηκε, είναι σημαντικό για την ανάπτυξη λογισμικού να υπάρχουν μέθοδοι αυτοματοποίησης όλων των φάσεων ελέγχου. Είναι δύσκολο να καθοριστεί το ακριβές ποσοστό οπτικών δοκιμών, καθώς συνήθως γίνονται παράλληλα με τις δοκιμές λειτουργικότητας της εφαρμογής, αλλά είναι βέβαιο πως αποτελούν εμπόδιο στην αυτονομία των αυτοματοποιημένων σουιτών, ενώ οι αλγοριθμικοί τρόποι σύγκρισης σχεδίων με στιγμιότυπων οθονών (εικονοστοιχείο προς εικονοστοιχείο) δεν επαρκούν. Συνεπώς, υπάρχει έλλειψη μεθόδων που επιτρέπουν τον πλήρη αυτοματοποιημένο έλεγχο της ιστοσελίδας σε όλα τα επίπεδα, με κύριο παράγοντα τους οπτικούς ελέγχους.

## 1.2 Στόχοι Αυτής της Διπλωματικής

Με την άνοδο των μοντέλων επεξεργασίας πολυμέσων και φυσικής γλώσσας, βασισμένα στη μηχανική μάθηση, ανοίγεται πλέον ένας καινούριος ορίζοντας στο χώρο ανάπτυξης λογισμικού γενικότερα και ειδικότερα στο χώρο δοκιμών. Εκεί οπού προηγουμένως θα έπρεπε να υπάρχουν περίπλοκοι, στατιστικοί / μαθηματικοί αλγόριθμοι πλέον υπάρχουν ευρέως διαθέσιμα εργαλεία τεχνητής νοημοσύνης ( μηχανικής μάθησης ), τα οποία έχουν πολύ υψηλές αποδόσεις στον συνδυασμό λεκτικών και οπτικών στοιχείων. Συγκεκριμένα, το μοντέλο της OpenAI για το ChatGPT, GPT-4o (o – omni), προσφέρει API για την ένταξή της σε εφαρμογές, ικανό να επεξεργαστεί φωτογραφίες, κείμενο, βίντεο και ήχο **[5]**. Ο σκοπός λοιπόν της εργασίας αυτής, είναι η ενσωμάτωση του εν λόγω μοντέλου σε πραγματική περίπτωση χρήσης  σε ένα μεγάλο και περίπλοκο υπό ανάπτυξη σύστημα. Οι κύριοι στόχοι είναι 2:

1) Ανάπτυξη σουίτας ελέγχου γραμμένη σε C# με οδηγό φυλλομετρητή Playwright (για πλοήγηση εντός της ιστοσελίδας και ανάληψη στιγμιότυπων οθόνης) και ενσωμάτωση ChatGPT-4o. Αυτή η σουίτα ελέγχου δεν θα περιέχει περίπλοκη διεπαφή χρήστη, αλλά

μόνο μια απλή διεπαφή αντιστοίχισης των σχεδίων Figma με το εκάστοτε βήμα του φυλλομετρητή.

2) Αξιολόγηση και εκτίμηση της ακρίβειας του εργαλείου αυτού, με την καταμέτρηση ορθών αποτελεσμάτων σε διάφορα είδη σφαλμάτων. Ταυτόχρονα, θα γίνει κατηγοριοποίηση μερικών βασικών ειδών σφαλμάτων (π.χ. τοποθέτηση, χρωματισμός, διάταξη, γραμματοσειρές) και βασική καταμέτρηση ορθών / λανθασμένων αποτελεσμάτων και ανάλυση αυτών.

Παράλληλα, θα γίνει μια ανασκόπηση βιβλιογραφίας σχετικά με αυτό το θέμα και μία σύντομη σύγκριση με δημοφιλή εργαλεία στη σημερινή αγορά, και θα ακολουθηθούν όλα τα βασικά βήματα ανάπτυξης εργαλείων λογισμικού.

## 1.3 Περίληψη Κεφαλαίων

Η παρούσα διπλωματική εργασία είναι δομημένη σε πέντε βασικά κεφάλαια και παραρτήματα, τα οποία καλύπτουν συνολικά τη θεωρητική βάση, την ανάλυση, την υλοποίηση και την αξιολόγηση της προτεινόμενης λύσης. Τα κεφάλαια έχουν ως εξής:

- **Κεφάλαιο 1:** Παρουσιάζει την εισαγωγή στο πρόβλημα του οπτικού ελέγχου, τους βασικούς στόχους της εργασίας και περιγράφει τη γενική δομή του εγγράφου.
- **Κεφάλαιο 2:** Περιλαμβάνει την ανασκόπηση της σχετικής βιβλιογραφίας, εξετάζοντας υπάρχουσες προσεγγίσεις βασισμένες σε χαρακτηριστικά εικόνας, DOM ανάλυση, CSS διαφορές και μεθόδους που αξιοποιούν τεχνητή νοημοσύνη.
- **Κεφάλαιο 3:** Αποτελεί το έγγραφο προδιαγραφών απαιτήσεων και περιγράφει αναλυτικά το ιστορικό εκδόσεων, τη μεθοδολογία που ακολουθήθηκε, καθώς και τις λειτουργικές, μη-λειτουργικές και επιχειρησιακές απαιτήσεις. Παρουσιάζεται επίσης η βασική σχεδίαση του συστήματος.
- **Κεφάλαιο 4:** Παρουσιάζει αναλυτικά την υλοποίηση της εφαρμογής για τον αυτόματο οπτικό έλεγχο ιστοσελίδων. Περιγράφεται η αρχιτεκτονική της λύσης, η τμηματοποίηση σε επίπεδα (GUI, Επιχειρησιακή Λογική, Πρόσβαση Δεδομένων), καθώς και οι βασικές λειτουργικές μονάδες του συστήματος. Τεκμηριώνονται οι κύριες ροές χρήσης, η αλληλεπίδραση των κλάσεων, καθώς και ο τρόπος με τον οποίο αξιοποιούνται το Playwright και το GPT-4o για την

παραγωγή και αξιολόγηση δοκιμών. Επιπλέον, παρουσιάζεται σχετικό διάγραμμα και αναλύονται οι δυνατότητες επέκτασης του συστήματος.

- **Κεφάλαιο 5:** Εστιάζει στην αξιολόγηση της εφαρμογής, παρουσιάζοντας μετρήσεις, συγκρίσεις και αποτελέσματα από την εκτέλεση των δοκιμών.
- **Κεφάλαιο 6:** Περιλαμβάνει τη συνολική επιχειρησιακή σύνοψη και τα βασικά συμπεράσματα της εργασίας, αναδεικνύοντας τη συνεισφορά και τα περιθώρια μελλοντικής επέκτασης.
- Στα **Παραρτήματα**, περιλαμβάνονται επιπλέον τεχνικές πληροφορίες όπως το εγχειρίδιο χρήστη, τεχνικό εγχειρίδιο, απαιτήσεις εργαλείων και ενδεικτικά δείγματα χρήσης.

Η δομή αυτή στοχεύει στη σταδιακή εμβάθυνση του αναγνώστη στο πρόβλημα, την τεκμηρίωση και την αξιολόγηση της υλοποιηθείσας λύσης.

# Κεφάλαιο 2



## Κεφάλαιο 2: Ανασκόπηση Βιβλιογραφίας

### 2.1 Γενικά

Στην αγορά υπάρχουν και υπήρξαν γενικά πολλά είδη εργαλείων βασισμένα σε διαφορετικές μεθόδους ανάλυσης των σχεδίων της ιστοσελίδας. Μάλιστα, συνολικά η παγκόσμια αγορά για αυτοματισμό ελέγχου υπολογίζεται γύρω στα 28.1 δισεκατομμύρια Αμερικανικά δολάρια για το έτος 2023 **[6]**. Μερικές από αυτές τις μεθόδους βασίζονται στην σύγκριση των εικόνων με βάση τα σχέδιά τους. Σύμφωνα με τον Μούς Χόνδα, τον Διευθυντή Ποιότητας στην Katalon, Inc. (που κατείχαν ~9% της παγκόσμιας αγοράς για αυτοματισμό δοκιμών Διεπαφής Χρήστη **[7]**) «ο οπτικός έλεγχος είναι μια τεχνική ελέγχου λογισμικού που περιλαμβάνει την λήψη στιγμιότυπων οθόνης της Διεπαφής Χρήστη και την σύγκριση αυτών με ένα σύνολο εικόνων αναφοράς για την ανίχνευση οποιωνδήποτε οπτικών διαφορών ή σφαλμάτων» **[8]**. Σύμφωνα με αυτήν την κατηγορία, λοιπόν, ο έλεγχος γίνεται συγκρίνοντας τις εικόνες του παρόντος συστήματος με τα πρότυπα που είχαν ήδη σχεδιαστεί από την ομάδα σχεδιαστών. Συνεπώς, μία κατηγορία δοκιμών είναι «ως προς την εικόνα». Βασίζεται στο αποτέλεσμα που όντως φαίνεται μετά την υλοποίηση του κώδικα οπτικά στον άνθρωπο, συγκρίνοντάς το αντίστοιχα, με την σχεδιασμένη εικόνα μέσω αυτοματοποιημένων μέσων.

Σύμφωνα όμως με την Smartbear ο οπτικός έλεγχος, είναι «μία μέθοδος ελέγχου λογισμικού, με την οποία επικυρώνεται η οπτική πιστότητα της Διεπαφής Χρήστη (UI) ή

Γραφικής Διεπαφής Χρήστη (GUI)». Κατά συνέπεια, σύμφωνα με αυτόν τον ορισμό, ο οπτικός έλεγχος είναι κάτι γενικότερο από σύγκριση μεταξύ εικόνων. Άρα, μπορούν να χρησιμοποιηθούν άλλες μέθοδοι πέραν της σύγκρισης εικόνων, σε επίπεδο κώδικα. Αυτό συμπεριλαμβάνει πόρους που χρησιμοποιούνται όπως τα Διαδοχικά Φύλλα Ύφους, και άρα είναι «ως προς την υλοποίηση». Για παράδειγμα, μέσω της χρήσης ενός οδηγού φυλλομετρητή όπως το Selenium, χρησιμοποιώντας την εντολή getCssValue("color"), υπάρχει η δυνατότητα να ελεγχθεί ότι όντως το χρώμα ενός στοιχείου αντιστοιχεί με τον εκάστοτε κωδικό RGB που είχε προκαθοριστεί από τις απαιτήσεις **[9]**. Αυτά τα εργαλεία συγκρίνουν μέσω μεθόδων κώδικα τις ιδιότητες των αντικειμένων, λ.χ. της ιστοσελίδας και την μια-προς-μια αντιστοίχιση αυτών των σχεδιαστικών στοιχείων. Συνήθως, είναι πιο εύκολο αυτό να γίνει παράλληλα και με τις δοκιμές λειτουργικότητας, καθώς πολλοί οδηγοί φυλλομετρητή προσφέρουν αυτές τις δυνατότητες.

Συμπερασματικά, αυτές οι δύο κύριες μεθοδολογίες, δηλαδή «ως προς την εικόνα» και «ως προς την υλοποίηση», είναι όροι «ομπρέλα» που θα μπορούσαν όμως να χωριστούν σε περισσότερες υποκατηγορίες όπως εξηγούνται παρακάτω, διότι ο τρόπος με τον οποίο προσεγγίζονται αυτές οι μεθοδολογίες εξαρτάται από τις τεχνικές υλοποίησης του εκάστοτε εργαλείου. Αυτό προκύπτει από την προσέγγιση των προγραμματιστών για την ανάλυση των εικόνων ή των στοιχείων του συστήματος. Όπως είναι γνωστό, υπάρχουν διάφορες μέθοδοι ανάλυσης εικόνας (αλγοριθμικοί ή με μέσα τεχνητής νοημοσύνης) όπως και διάφορες προσεγγίσεις ανάλυσης της υλοποίησης του συστήματος. Αναπόφευκτα, εν συνεχεία, αναλόγως της μεθοδολογίας υπάρχουν και αντίστοιχες αδυναμίες ή πλεονεκτήματα. Παρακάτω, θα γίνει σύντομη αναφορά των πιο δημοφιλών μεθόδων στην τωρινή αγορά, καθώς και παρουσίαση των βασικών τους χαρακτηριστικών.

## 2.2 Ανάλυση με βάση τα στοιχεία της εικόνας.

Αυτή η μέθοδος είναι κατ' εξοχήν «ως προς την εικόνα». Η βασική ιδέα πίσω από αυτήν την μέθοδο είναι απλή· δεδομένης της βασικής ή αναμενόμενης εικόνας, γίνεται σύγκριση σε επίπεδο πρώτα των εικονοστοιχείων μεταξύ των εικόνων όπου και ανιχνεύονται οι διαφορές, μετέπειτα στην διάταξη και ομαδοποίησή τους (σχήματα) και εν τέλει κατά το περιεχόμενό τους. Σύμφωνα με το εγχειρίδιο της Katalon **[10]**, η μέθοδος της σουίτας τους είναι αυτή. Ομοίως, η δυνατότητα οπτικών ελέγχων της LambdaTest,

παρουσιάζει την διαφορά μεταξύ εικονοστοιχείων της εικόνας αναφοράς του στιγμιότυπου οθόνης της εκάστοτε σελίδας **[11]**.

Προφανώς, ένα κύριο πλεονέκτημα αυτής της μεθόδου είναι πως είναι απλός ο τρόπος λειτουργίας της. Στην περίπτωση όπου γίνεται σύγκριση ακριβώς της ίδιας έκδοσης, και όπου οι συνθήκες λειτουργίας του συστήματος είναι καλά καθορισμένες, μπορεί να υπάρξει πολύ καλή ακρίβεια στα αποτελέσματα. Δύναται να βρει την πιο μικρή διαφορά πολύ πιο εύκολα από τον άνθρωπο, καθώς ειδικά σε μεγάλα συστήματα, είναι εύκολο να διαφύγουν οι μικρές λεπτομερείς σε έναν άνθρωπο. Ταυτόχρονα, καθώς βασίζεται στην εύρεση διαφορών μεταξύ των εικονοστοιχείων, είναι πολύ δύσκολο να αστοχήσει όσον αφορά υπαρκτά λάθη, καθώς είναι μέθοδος υψηλής ακριβείας.

Όμως, από την άλλη, επειδή ακριβώς βασίζονται σε πολύ συγκεκριμένες τοποθετήσεις στοιχείων στην εικόνα, είναι πολύ πιθανόν να υπάρξει μεγάλος αριθμός λανθασμένων θετικών αποτελεσμάτων για σφάλματα, δηλαδή να μεταφράσει απλές μεταβολές μεταξύ διαφόρων τοποθετήσεων ή διαστάσεων σε σφάλματα, μόνο και μόνο διότι υπάρχουν μικρές διαφορές σε εικονοστοιχεία. Αυτό αναγκάζει τον υπεύθυνο ελέγχου να προβεί ο ίδιος με την σειρά του σε μεγάλο ποσοστό επικυροποίησης των αποτελεσμάτων του αυτόματου ελέγχου, μειώνοντας σε μεγάλο βαθμό την αποτελεσματικότητά του. Ταυτόχρονα, καθώς επεξεργάζεται πόρους εικόνας εικονοστοιχείο προς εικονοστοιχείο, θα μπορούσε να θεωρηθεί σχετικά ακριβό ως προς τις ανάγκες επεξεργαστικού χρόνου, καθώς αναγκαστικά «περνά» πάνω από όλη την εικόνα, ακόμα και σημεία που πιθανώς να είναι αδιάφορα.

## 2.3 Χρήση Μοντέλων Αντικειμένου Εγγράφου (DOM) και διαφορές σε Διαδοχικά Φύλλα Ύφους (CSS)

Η βασική μέθοδος «ως προς την υλοποίηση» όσον αφορά στα συστήματα διαδικτύου ή εφαρμογές βασίζεται στην ανάλυση των περιεχομένων DOM και αντίστοιχα στο CSS. Μέσω βεβαιώσεων, μπορούν να συγκριθούν οι προκαθορισμένες προδιαγραφές σχεδίασης με αυτές που ανιχνεύουν οι οδηγοί φυλλομετρητή σε επίπεδο κώδικα, καθώς εν τέλει σε ιδανικές περιπτώσεις ανάπτυξης λογισμικού είναι γνωστές στους ελεγκτές οι αναμενόμενες ιδιότητες.

Για παράδειγμα, στο Selenium WebDriver **[12]** υπάρχουν οι εξής εντολές:

```
WebDriver driver = new ChromeDriver();
```

```
driver.findElement(By.cssSelector("#fname"));

driver.findElement(By.tagName("div")).getAttribute("style");
```

Αυτές, εντοπίζουν στοιχεία βάση των ιδιοτήτων τους στο CSS, την κατηγορία στην οποία ανήκουν, καθώς και τον εντοπισμό των στυλ που είχαν προκαθοριστεί. Μέσω ελέγχων λοιπόν και την χρήση των ελέγχων δοκιμών της εκάστοτε γλώσσας προγραμματισμού μπορεί να γίνει σύγκριση με τα επιθυμητά στυλ/χρώματα και άλλα χαρακτηριστικά.

Ομοίως χρησιμοποιώντας άλλον οδηγό φυλλομετρητή, τον Cypress, κάποιος μπορεί να συντάξει το εξής παράδειγμα ελέγχου **[13]**:

```
cy.get('.completed').should('have.css', 'text-decoration', 'line-through')
cy.get('.completed').should('have.css', 'color', 'rgb(217,217,217)')
```

Αυτός παραδείγματος χάριν, είναι έλεγχος που ελέγχει τις ιδιότητες του CSS, σε διακόσμηση κειμένου και σε χρώμα.

Συνοπτικά, αυτή η μέθοδος έχει τις εξής ιδιότητες· πρώτον, βασίζεται στην ικανότητα του εκάστοτε οδηγού φυλλομετρητή να εντοπίζει τις πληροφορίες του σχεδιασμού ενός αντικειμένων και δεύτερον, βασίζεται στην «εξυπνάδα» του ελεγκτή λογισμικού στο κατά πόσο θα χρησιμοποιήσει τα κατάλληλα τεχνάσματα που απαιτούνται από το εκάστοτε πρόβλημα. Αυτό σημαίνει πως υπάρχει περιορισμός στο εύρος και βάθος ελέγχου που μπορεί να γίνει, καθώς και απαίτηση συνεχών αλλαγών στον κώδικα όταν προκύπτουν αλλαγές στις σχεδιαστικές προδιαγραφές. Ταυτόχρονα, κατά τον σχεδιασμό ενός συστήματος, δεν καθορίζονται πάντοτε όλα τα στυλ με πάσα λεπτομέρεια, καθώς υπάρχουν βιβλιοθήκες σχεδιασμού υψηλότερου επιπέδου. Αυτό σημαίνει πως τοποθετείται επιπλέον «φόρτος» καθορισμού των αναμενόμενων αποτελεσμάτων στον ελεγκτή/προγραμματιστή.

Ταυτόχρονα όμως, το κόστος επεξεργασίας είναι αναλογικά χαμηλό σε σχέση με την επεξεργασία εικόνας. Δεν απαιτείται ο χρονοβόρος έλεγχος της δομής της εικόνας, αλλά ο σχετικά γρήγορος χρόνος πρόσβασης των Μοντέλων Αντικειμένου Εγγράφου (e.g. HTML) η των Διαδοχικών Φύλλων Ύφους (CSS). Επίσης σε περιπτώσεις βασικών δοκιμών που δεν είναι περίπλοκες, μπορούν να προσφέρουν μια σχετικά απλή μέθοδο επικύρωσης η οποία έχει υψηλή ακρίβεια.

## 2.4 Οπτικοί Έλεγχοι ενισχυμένοι με Τ.Ν. (Τεχνητή Νοημοσύνη)

Αυτή η μέθοδος είναι κατά κύριο λόγο «ως προς την εικόνα». Τα εργαλεία που χρησιμοποιούν αυτήν την μέθοδο, χρησιμοποιούν εργαλεία τεχνητής νοημοσύνης, και πιο συχνά μηχανικής μάθησης, τα οποία έχουν εκπαιδευτεί στην αναγνώριση εικόνων. Κάποια από τα εργαλεία που χρησιμοποιούν αυτές τις μεθόδους έχουν ως εξής· VisualTest της SmartBear **[14]**, το οποίο σε αντίθεση με την σύγκριση εικονοστοιχείου προς εικονοστοιχείου προσθέτει εργαλείο Τ.Ν. το οποίο «σκαννάρει» τις εικόνες, για την μείωση των ψευδών θετικών ευρέσεων σφάλματος, το Functionize **[15]** το οποίο χρησιμοποιεί μεθόδους μηχανικής μάθησης και υπολογιστικής όρασης ούτως ώστε να προσαρμόζεται σε μικρές αλλαγές μεταξύ των διαφορετικών συσκευών και οθονών κ.α. Ταυτόχρονα, εργαλεία που αναφέρθηκαν στις μεθόδους εικονοστοιχείο-προς-εικονοστοιχείο, επίσης μπορεί να χρησιμοποιήσουν βελτιώσεις μέσω τεχνητής νοημοσύνης βελτίωση της επίδοσής τους.

Εν γένει, τυγχάνουν χρήσης διαφορετικά μοντέλα Τ.Ν., είτε εμπίπτουν στην κατηγορία της μηχανικής μάθησης είτε όχι. Λόγου χάριν, μέσω αλγορίθμων υπολογιστικής όρασης, μοντέλα όπως το Applitools Visual AI υποβοηθούνται να αναλύουν εικόνες και να εντοπίσουν ασυνέπειες στη Διεπαφή Χρήστη, «κατανοώντας» το συγκείμενο των στοιχείων αντί ωμής διαφοράς εικονοστοιχείων. Ταυτόχρονα, όσον αφορά την μηχανική μάθηση χρησιμοποιούνται και άλλες μέθοδοι. Τα Συνελικτικά Νευρονικά Δίκτυα (CNNs) χρησιμοποιούνται για ανίχνευση μη-προσδοκώμενων διαφορών στην διάταξη και να ιχνηλατούν αντικείμενα μεταξύ οθονών, όπως στο Functionize AI. Τα μοντέλα Επεξεργασίας Φυσικής Γλώσσας (NLP) βοηθούν την ανίχνευση αλλαγών σε κείμενο ενώ η ενισχυτική μάθηση βοηθά τα μοντέλα αυτά να λαμβάνουν πιο σωστές επιλογές.

Το δε εργαλείο υπό ανάπτυξη σε αυτήν την διπλωματική, εμπίπτει σε αυτήν την κατηγορία. Χρησιμοποιώντας το μοντέλο Chat-GPT 4ο προσπαθεί να αξιοποιήσει τις πολυτροπικές ικανότητές του (κείμενο, εικόνα) καθώς και την υψηλή κατανόηση γλώσσας (ως LLM), για τον εντοπισμό αυτών των σφαλμάτων. Είναι επίσης καλό στο να γενικεύει από λίγα παραδείγματα (Few-Shot learning), κάτι σημαντικό για τα οπτικά σφάλματα. Παράλληλα, έχει ενσωματωμένη μέθοδο ενισχυτικής μάθησης μέσω

ανατροφοδότησης από άνθρωπο, που σημαίνει πως μπορεί να εκπαιδευτεί με σχετικά πιο εύκολο τρόπο αν τυχόν χρειαστεί.

Επιπλέον, το μοντέλο αυτό έχει δείξει υψηλά αποτελέσματα ακριβείας σε διάφορες περιπτώσεις αναγνώρισης εικόνων. Σε αναγνώριση ιατρικών εικόνων, σε ερωτήσεις από τις εξετάσεις πιστοποίησης ιατρικής στις ΗΠΑ **[16]** (United States Medical Licensing Examination), πέτυχε ακρίβεια 70.8%, 92.9% και 62.5% στα τρία της μέρη, έχοντας ποσοστό επιτυχίας στην εξέταση. Σε άλλη έρευνα, το μοντέλο ανέδειξε υποσχόμενες δυνατότητες στην αναγνώριση ζωικών συμπεριφορών μέσω οπτικού υλικού, ακόμα και βίντεο **[17]**. Αυτά καθιστούν το μοντέλο καλή επιλογή για οπτικό έλεγχο.

## 2.5 Τρόπος Αξιολόγησης

Το ερώτημα που προκύπτει είναι, δεδομένων όλων αυτών των μεθόδων, ποια είναι η πιο αποτελεσματική; Εφόσον μιλούμε για εύρεση σφαλμάτων, θα πρέπει να χρησιμοποιήσουμε κάποια μετρικά προσμέτρησης για αυτά. Ο πιο λογικός και κατανοητός, αλλά επίσης ευρέως διαδεδομένος τρόπος, είναι η χρήση της ανάκλησης (Recall) και της ακρίβειας (Precision) ως ορίζονται κάτωθεν:

$$\boldsymbol{Ακρίβεια} = \frac{\boldsymbol{Ορθά\ Θετικά}}{\boldsymbol{Ορθά\ Θετικά} + \boldsymbol{Λανθασμένα\ Θετικά}}$$

$$\boldsymbol{Ανάκληση} = \frac{\boldsymbol{Ορθά\ Θετικά}}{\boldsymbol{Ορθά\ Θετικά} + \boldsymbol{Λανθασμένα\ Αρνητικά}}$$

Όπου τα ορθά θετικά (ΟΘ) είναι η σωστή ανίχνευση ενός πραγματικού σφάλματος, τα λανθασμένα θετικά (ΛΘ) είναι η ανίχνευση ενός σφάλματος το οποίο όμως δεν υπάρχει και αντιστοίχως τα αρνητικά (πραγματική μη-εύρεση σφάλματος έναντι μη-εύρεση σφάλματος το οποίο όμως υπάρχει).

Σε όρους σχετικούς με το εξής πρόβλημα, η ακρίβεια μετράει ποσά από τα αναφερόμενα ελαττώματα είναι πραγματικά. Πιο υψηλή ακρίβεια σημαίνει πως το εργαλείο είναι πιο αξιόπιστο. Η δε ανάκληση, μετρά αντίθετα την ικανότητα του εργαλείου να εντοπίσει *όλα* τα ελαττώματα, δηλαδή πόσο εύκολα εντοπίζει κάθε είδος ελαττώματος, ελαχιστοποιώντας τις παραβλέψεις. Παράλληλα, αυτές οι μετρήσεις πρέπει να γίνουν σε διαφορετικές κατηγορίες. Οι βασικές είναι κείμενο (γραμματοσειρά), αντικείμενα (σχήματα), χρώμα και διάταξη (τοποθέτηση αυτών στο εκάστοτε πλαίσιο. Συνεπώς, για

σκοπούς αξιολόγησης του εργαλείου θα χρησιμοποιηθούν πραγματικά δείγματα από κάθε κατηγορία.

Για καλύτερη όμως γενική εικόνα της ποιότητας, με βάση τα δυο πιο πάνω μετρικά θα χρησιμοποιηθεί και ο παράγοντας F-Score.

$$\boldsymbol{F = 2 * \frac{Ακρίβεια * Ανάκληση}{Ακρίβεια + Ανάκληση}}$$

Αυτό εκφράζει την ισορροπία μεταξύ ανάκλησης και ακρίβειας σε περίπτωση που υπάρχει άνιση διανομή αυτών.

Γενικώς, αυτοί οι μέθοδοι μέτρησης είναι που χρησιμοποιούνται για εργαλεία AI. Σύμφωνα με την έρευνα "Artifical intelligence for context-aware visual change detection in software test automation", η μέθοδος βαθιάς μάθησης (deep learning) που χρησιμοποιήσαν μπορούσε να ξεπεράσει σε απόδοση τα τυπικά εργαλεία δοκιμών, με υψηλή ακρίβεια (μέχρι και 93% σε συγκεκριμένες κατηγορίες) **[18]**. Αξιολογώντας λοιπόν αυτό το εργαλείο, θα εξερευνηθεί η χρηστικότητά του σε αυτόν τον τομέα, καθώς και η πιθανή του μελλοντική ανάπτυξη για αυτονομία του αυτόματου ελέγχου για οπτικές δοκιμές. Είναι δύσκολο, όμως, να συγκριθεί με άλλα εργαλεία που υπάρχουν ήδη στην αγορά, καθώς δεν εξάγουν επακριβώς τα αποτελέσματά τους.

# Κεφάλαιο 3



## Κεφάλαιο 3: Έγγραφο Προδιαγραφών Απαιτήσεων

### 3.1 Ιστορικό Εκδόσεων

| Έκδοση # | Εφαρμόστηκε Από | Ημ/νια αναθεώρησης | Εγκρίθηκε από | Ημ/νια έγκρισης | Λόγος |
|---|---|---|---|---|---|
| 1.00 | Ιάσων Γεωργίου | 02/03/2025 | Νίκος Αναστασίου | 4/03/2025 | Ελλιπή σχεδιαγράμματα |
| 2.00 | Ιάσων Γεωργίου | 12/03/2025 | Νίκος Αναστασίου | 17/03/2025 | Συμφωνία μη-λειτουργικών απαιτήσεων |
| 3.00 | Ιάσων Γεωργίου | 13/04/2025 | Νίκος Αναστασίου | 04/21/2025 | Καθορισμός απαιτήσεων για Chat-GPT 4o tokens |

## 3.2 Εισαγωγή

Το έργο αυτό έχει ως σκοπό την εξερεύνηση των δυνατοτήτων του εργαλείου Chat-GPT 4o στην διάγνωση οπτικών σφαλμάτων ενός διαδικτυακού συστήματος ως προς τα σχέδια αυτού (π.χ. Figma) μέσω μιας εφαρμογής. Ως εκ τούτου, απαιτείται η δημιουργία μιας διασύνδεσης σε πράκτορα Chat GPT-4o, με μια απλή διεπαφή χρήστη για χρήση, καταγραφή δοκιμών, διαχείριση δοκιμών και παραγωγής αποτελεσμάτων για την εκτίμησή τους. Θα πρέπει ο πηγαίος κώδικας της εφαρμογής να είναι γραμμένος σε C# ούτως ώστε να έχει εύκολη διασύνδεση με τον δυνατό πυρήνα δοκιμών της .NET, όπως και του δυνατού οδηγού φυλλομετρητή Playwright, με τέτοιον τρόπο που να μπορεί μελλοντικά να τροποποιηθεί σε πλήρες σύστημα διαχείρισης δοκιμών.

### *3.2.1 Σκοπός*

Ο σκοπός του εγγράφου αυτού είναι να ορίσει τις λειτουργικές και μη-λειτουργικές απαιτήσεις για το Εργαλείο Αυτοματοποιημένου Οπτικού Ελέγχου, μια εφαρμογή λογισμικού γραμμένη στην C# ως μέρος μιας διπλωματικής εργασίας με τον Ιάσονα Γεωργίου σε συνεργασία με την εταιρεία TechLink. Το εργαλείο αυτό είναι σχεδιασμένο να βοηθήσει τους προγραμματιστές και ελεγκτές λογισμικού να ανιχνεύσουν λανθασμένες οπτικές διαφορές κατά την διάρκεια του ελέγχου διαδικτυακών ή άλλων παρόμοιων συστημάτων. Το εργαλείο αναπτύσσεται σε επαγγελματικό περιβάλλον υπό την συνεπίβλεψη του επιβλέποντα καθηγητή διπλωματικής εργασίας Δρ. Κωνσταντίνο Παττίχη και τον Διευθυντή QA κ. Νίκο Αναστασίου.

### *3.2.2 Πεδίο Εφαρμογής*

Το εργαλείο αυτό είναι γραμμένο στη C# με διασύνδεση σε πράκτορα Chat GPT-4o και την χρήση του Playwright. Θα έχει την δυνατότητα να λαμβάνει στιγμιότυπα οθόνης, σε σενάρια δοκιμών που θα καταγράφονται (recording) από τον χρήστη. Το αποτέλεσμα θα είναι η δημιουργία μιας αναφοράς HTML, όπου καταγράφονται οι διαφορές μεταξύ των στιγμιότυπων και των σχεδίων της ιστοσελίδας. Η βάση του κώδικα θα πρέπει να είναι συμβατή με το NUnit. Δεν θα περιλαμβάνει λειτουργικούς ελέγχους της ιστοσελίδας, ή ελέγχους απόδοσης κ.ο.κ.. Αφορά τους μηχανικούς QA αλλά και τους μηχανικούς λογισμικού που αναπτύσσουν σύστημα διαδικτύου για τον έλεγχο του προϊόντος τους.

### *3.2.3 Υπόβαθρο*

Το έγγραφο παράγεται ως μέρος μιας διπλωματικής εργασίας προπτυχιακού επιπέδου σε συνεργασία με τον επιβλέποντα καθηγητή Δρ. Κωνσταντίνο Παττίχη, τον φοιτητή και μηχανικό QA κ. Ιάσων Γεωργίου, και τον διευθυντή QA της εταιρείας TechLink. Ο στόχος του έργου είναι η σχεδίαση και η ανάπτυξη ενός εργαλείου που αυτοματοποιεί τον επαναληπτικό οπτικό έλεγχο για λογισμικά.

Η ανάπτυξη αυτού του συστήματος τυγχάνει επίβλεψης και από ακαδημαϊκή άποψη και επαγγελματική. Το έργο αυτό ανταποκρίνεται σε μια πραγματική ανάγκη της τρέχουσας αγοράς λογισμικού αλλά και εντός του οργανισμού, για πιο αξιόπιστη και αποτελεσματική διαδικασία ελέγχου της διεπαφής χρήστη, ιδιαιτέρως όσον αφορά στην συνεχή διασύνδεση και ανάπτυξη (δηλαδή στις επαναληπτικές δοκιμές σε διάφορες φάσεις ανάπτυξης, όταν αυτό απαιτείται).

Ο οργανισμός TechLink έχει ευθύνη για την επικύρωση και την εφαρμογή του εργαλείου εντός του κύκλου ανάπτυξης του λογισμικού της, αφού μελετήσει τα αποτελέσματα εκτιμήσεων με το πέρας αυτής της εργασίας, όπως περιγράφονται εκτός του εγγράφου των απαιτήσεων, καθώς δεν υπάρχει συγκεκριμένη απαιτούμενη ακρίβεια ή απόδοση επί του παρόντος. Ταυτόχρονα, προσφέρει χώρο στο νέφος για την ανάπτυξη του λογισμικού και δείγματα σχεδίων ιστοσελίδας για την εκτίμηση της απόδοσης του εργαλείου. Ο καθηγητής Δρ. Κωνσταντίνος Παττίχης προσφέρει καθοδήγηση για μεθοδολογία έρευνας, τεκμηρίωση εγγραφών και το τεχνικό βάθος που απαιτείται για την επιτυχημένη ολοκλήρωση της διπλωματικής.

### *3.2.4 Αναφορές*

#### 3.2.4.1 Τεχνικές Αναφορές

- Περιβάλλον δοκιμών για την C# NUnit: https://nunit.org/
- Οδηγός Φυλλομετρητή Playwright: https://playwright.dev/
- Δυνατότητες Καταγραφής Δοκιμών Playwright στο NUnit C#: https://playwright.dev/dotnet/docs/codegen-intro
- Περιβάλλον Azure για την χρήση της εγκατάστασης πράκτορα Chat-GPT 4o: https://learn.microsoft.com/en-us/azure/ai-foundry/what-is-azure-ai-foundry (χώρος προσφερόμενος από τον Οργανισμό TechLink)

- Αποθετήριο κώδικα και χώρο εργασίας εντός του Azure: https://azure.microsoft.com/en-us/products/devops/repos (χώρος προσφερόμενος από τον Οργανισμό TechLink)

#### *3.2.4.2 Σχετικά Έγγραφα*

1. Ενδεικτικά σχέδια Figma όμοια με αυτά υπάρχοντος συστήματος (ΤΟΜ – Τμήμα Οδικών Μεταφορών) όπου θα γίνει η εκτίμηση αποτελεσμάτων (προσφερόμενα από τον Οργανισμό TechLink).
2. Δείγματα για διαφορετικά είδη οπτικών σφαλμάτων τροποποιημένα από το (1).

### *3.2.5 Προϋποθέσεις και Περιορισμοί*

Αυτός ο τομέας ταυτοποιεί τις προϋποθέσεις και τους περιορισμούς που επηρεάζουν την ανάπτυξη και παράδοση των Λειτουργικών και Μη-Λειτουργικών Απαιτήσεων.

#### 3.2.5.1 Προϋποθέσεις

- Οι απαιτούμενες τεχνικές υποδομές (πλατφόρμα νέφους Azure, εγκαταστάσεις πράκτορα μοντέλου Τ.Ν.) θα είναι διαθέσιμες και λειτουργικές πριν την διαδικασία ανάπτυξης
- Όλοι οι ενδιαφερόμενοι (επιβλέπων καθηγητής, ο πελάτης-οργανισμός TechLink) θα είναι διαθέσιμοι για ανατροφοδότηση όταν αυτό απαιτείται.
- Δεν θα υπάρξουν κύριες διαφοροποιήσεις κατά την περίοδο ανάπτυξης
- Τα APIs τρίτων ή υπηρεσίες που χρησιμοποιούνται θα παραμείνουν σταθερές καθ’ όλη την διάρκεια του έργου.

#### 3.2.5.2 Περιορισμοί

- Ο πηγαίος κώδικας θα παραμείνει κλειστός καθώς τα πνευματικά του δικαιώματα ανήκουν στον προγραμματιστή και στον οργανισμό πελάτη (TechLink)
- Θα χρησιμοποιηθούν μόνο τεχνολογικά εργαλεία που εγκρίνονται από τον οργανισμό πελάτη
- Η εργασία πρέπει να ολοκληρωθεί με το πέρας του Εαρινού Ακαδημαϊκού Εξαμήνου 2025.

- Οποιαδήποτε αλλαγή μετά την εκκίνηση της φάσης ανάπτυξης πρέπει να έχει επίσημη έγκριση.

### *3.2.6 Περίληψη Εγγράφου*

1. Εισαγωγή: Παραχωρεί τον σκοπό, τους ορισμούς και τα ενδιαφερόμενα μέρη του εγγράφου.
2. Μεθοδολογία: Περιγράφεται η γενική προσέγγιση για το πως εξάγονται τα συμπεράσματα για τις λειτουργικές και μη λειτουργικές απαιτήσεις.
3. Λειτουργικές Απαιτήσεις: Περιγράφονται συγκεκριμένες λειτουργίες που πρέπει να εκτελεί η εφαρμογή.
4. Μη Λειτουργικές Απαιτήσεις: Αναφέρει την χρηστικότητα και άλλα στοιχεία.
5. Παραρτήματα

## 3.3 Μεθοδολογία

### *3.3.1 Γενική Προσέγγιση*

Η γενική προσέγγιση για την εξαγωγή των λειτουργικών και μη-λειτουργικών απαιτήσεων συμπεριλάμβανε τον συνδυασμό συμμετοχής των ενδιαφερόντων και επαναληπτική επικύρωση από αυτούς. Αρχικά, μια σειρά συνεντεύξεων έλαβε χώρο με τους κύριους ενδιαφερόμενους όπως αυτοί αναφέρονται στο Κεφάλαιο 1, και άλλους όπως μηχανικούς QA για τον καθορισμό τον τρεχόντων προκλήσεων στο χώρο του ελέγχου λογισμικού. Επιπρόσθετα, χρησιμοποιήθηκαν υπάρχοντα έγγραφα για τον καθορισμό του εύρους δεδομένων που θα μπορούν να χρησιμοποιηθούν για την επικύρωση της εφαρμογής αυτής.

Ταυτόχρονα, έγινε εξερεύνηση άλλων εργαλείων στα πλαίσια της βιβλιογραφικής μελέτης της διπλωματικής εργασίας, για τον καθορισμό των μετρικών της εκτίμησης της ακρίβειας της εφαρμογής αυτής, αλλά και για την κατηγοριοποίηση του εύρους των οπτικών σφαλμάτων που θα εξετάζει η εφαρμογή.

### *3.3.2 Μέθοδοι Μοντελοποιήσης*

Για να υποβοηθηθεί η ανάπτυξη λογισμικού και η κατανόηση του συστήματος από όλους τους ενδιαφερόμενους, θα χρησιμοποιηθούν διάφορα μοντέλα:

- Διαγράμματα Περίπτωσης Χρήσης (Use Case Diagrams): Αναπαράσταση διάδρασης του χρήστη με την εφαρμογή δοκιμών σε υψηλό επίπεδο
- Διαγράμματα Δραστηριότητας (Activity Diagrams): Για την ακολουθία των λειτουργιών στη δημιουργία δοκιμών και εκτέλεσης
- Διαγράμματα ροής (Flowcharts): Λογική της εφαρμογής

Επίσης εάν υπάρχει απαίτηση, μπορεί να εισαχθούν σε οποιοδήποτε μέρος οποιαδήποτε άλλα διαγράμματα μπορούν να βοηθήσουν στην ανάπτυξη και τον σχεδιασμό της εφαρμογής.

### *3.3.3 Εργαλεία και Τεχνολογίες*

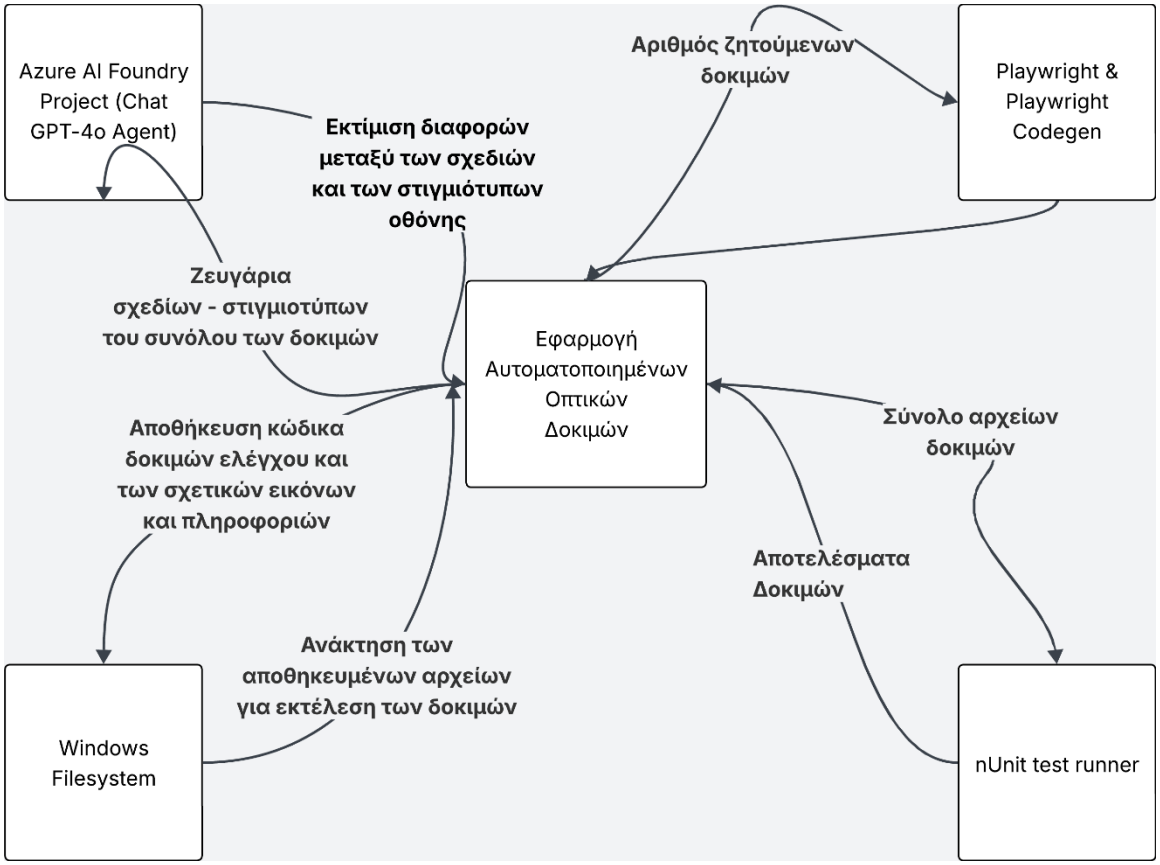
Η εφαρμογή θα πρέπει να βασίζεται στις ακόλουθες τεχνολογίες:

- **Visual Studio:** Θα χρησιμοποιηθεί για τις δυνατότητές του ενσωμάτωσης με τα αποθετήρια κώδικα στο Azure της εταιρείας και του μοντέλου AI
- **nUnit:** Θα χρησιμοποιηθεί ως πλαίσιο ανάπτυξης (Framework) των δοκιμών, ούτως ώστε να είναι εύκολα ενσωματοποιήσιμες με το γενικότερο πλαίσιο ανάπτυξης που χρησιμοποιείται από τον οργανισμό, επαναχρησιμοποιήσιμες και ευκόλως κατανοητές για προγραμματιστές με τη C#.
- **Playwright:** Ώστε να βασίζονται οι δοκιμές ελέγχου σε έναν σύγχρονο οδηγό φυλλομετρητή με πολλές δυνατότητες ο οποίος επίσης βρίσκεται σε χρήση από τον οργανισμό TechLink αλλά και παράλληλα από μεγάλο μερίδιο της αγοράς. Επίσης θα είναι εύκολο να τυγχάνει εκμετάλλευσης το εργαλείο Playwright Codegen, το οποίο χρησιμοποιείται στην παραγωγή κώδικα δοκιμών και είναι εύχρηστο για την απαίτηση της καταγραφής δοκιμών
- **Figma:** Για την χρήση των δειγμάτων σχεδίων του διαδικτυακού συστήματος. Θα χρησιμοποιηθούν ως η βάση δημιουργίας δειγμάτων βάσει των οποίων θα γίνει η εκτίμηση του εργαλείου.

- **Azure AI Foundary / Azure OpenAI:** Πλατφόρμα μέσα στην οποία θα γίνει η ανάπτυξη του πράκτορα AI Chat GPT-4o για την χρήση του από την εφαρμογή, καθώς αποτελεί το σύστημα DevOps της εταιρείας, και έχει καλή ενσωμάτωση με την γλώσσα C# όπως και με τα υπόλοιπα προαναφερθέντα εργαλεία.

### *3.3.4 Πλαίσιο Εφαρμογής*

Παρουσιάζεται το διάγραμμα πλαισίου της εφαρμογής, που αναδεικνύει τις βασικές διεπαφές με τις οποίες διασυνδέεται για την επίτευξη των σκοπών της




*Εικόνα 1 Διάγραμμα Πλαισίου*

## 3.4 Απαιτήσεις Δεδομένων Και Χρήστη

### *3.4.1 Απαιτήσεις Χρήστη*

Σε αυτό το τμήμα περιγράφονται οι απαιτήσεις των τελικών χρηστών της εφαρμογής. Οι κύριοι χρήστες της εφαρμογής είναι μηχανικοί QA (Quality Assurance), προγραμματιστές δοκιμών αυτοματοποίησης και οι ενδιαφερόμενοι στην ανάπτυξη της

εφαρμογής οι οποίοι αλληλοεπιδρούν με την εφαρμογή αυτοματοποιημένου οπτικού ελέγχου. Αυτές οι απαιτήσεις αντανακλούν την αναμενόμενη συμπεριφορά του συστήματος από την άποψη του χρήστη και επικεντρώνονται στην χρηστικότητα, προσβασιμότητα και στις λειτουργικές απαιτήσεις. Η εφαρμογή πρέπει να επιτρέπει τους χρήστες να δημιουργούν, να τρέχουν και να διαχειρίζονται αυτοματοποιημένους ελέγχους διεπαφής χρήσης για σκοπούς αξιολόγησης και προκαταρτικής χρήστης του εργαλείου.

Οι απαιτήσεις χρήστη έχουν ως εξής:

ΑΧ-1: Ο χρήστης θα πρέπει να είναι σε θέση να αλληλοεπιδρά με την εφαρμογή μέσω Γραφικής Διεπαφής Χρήστη (Graphical User Interface)

ΑΧ-2: Θα πρέπει να υπάρχει δυνατότητα ονοματοδοσίας των σεναρίων ελέγχου

ΑΧ-3: Θα πρέπει να υπάρχει δυνατότητα καταγραφής και δημιουργίας των σεναρίων ελέγχου

ΑΧ-4: Θα πρέπει να υπάρχει δυνατότητα αντικατάστασης των υπαρχόντων σεναρίων ελέγχου

ΑΧ-5: Θα πρέπει ο χρήστης να μπορεί μέσω Γραφικής Διεπαφής Χρήστη να τρέξει τα σενάρια ελέγχου

ΑΧ-6: Θα πρέπει ο χρήστης να μπορεί να συσχετίσει τα στιγμιότυπα οθόνης των σεναρίων ελέγχου με τα σχέδια βάσης

ΑΧ-7: Θα πρέπει ο χρήστης να μπορεί να ενημερώσει την συσχέτιση των στιγμιότυπων οθόνης σεναρίων και σχεδίων βάσης

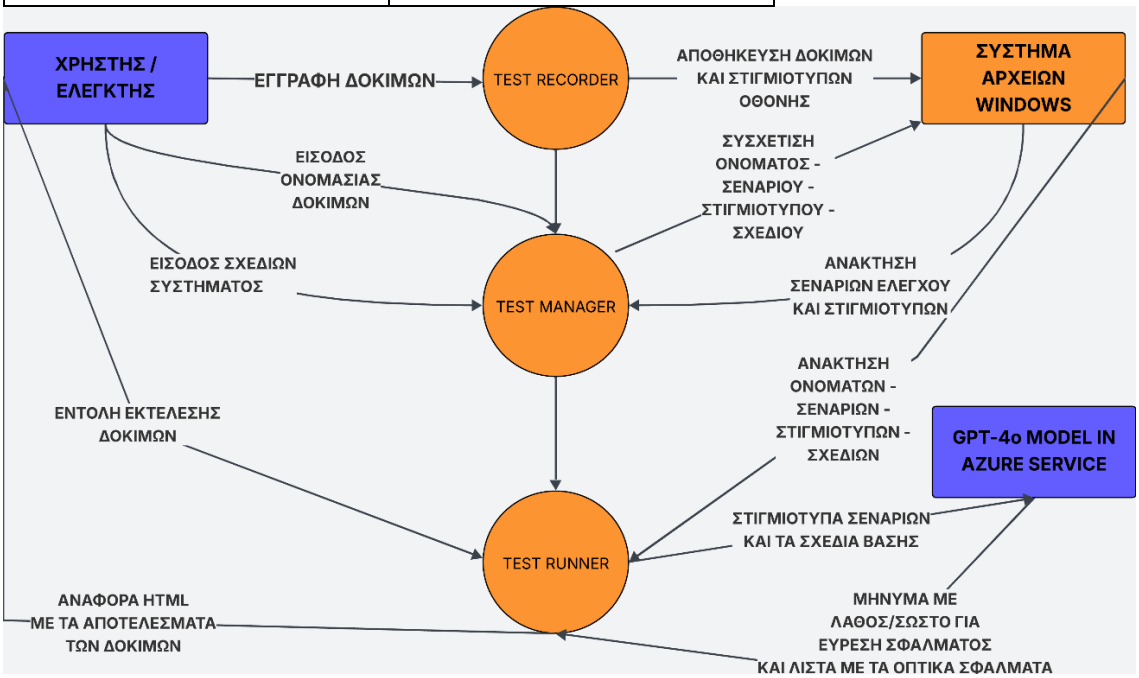
ΑΧ-8: Θα πρέπει ο χρήστης να μπορεί να δει μια αναφορά HTML των αποτελεσμάτων των σεναρίων

ΑΧ-9: Θα πρέπει ο χρήστης να μπορεί να εκκινήσει την εφαρμογή μέσω του αποθετηρίου κώδικα στο οποίο αναπτύχθηκε

### *3.4.2 Διάγραμμα Ροής Δεδομένων*

Στο διάγραμμα ροής δεδομένων, γίνεται μια σύντομη αλλά σαφής επεξήγηση της εφαρμογής σε πρώτο επίπεδο. Αποσυνθέτει την κύρια εφαρμογή στις διαδικασίες – κλειδιά της, την εγγραφή δοκιμών, την διαχείριση δοκιμών και την εκτέλεση δοκιμών. Οι εξωτερικές οντότητες όπως είναι ο Χρήστης και ο πράκτορας Chat-GPT 4o αλληλοεπιδρούν με την εφαρμογή μέσω των διαδικασιών που προαναφέρθηκαν, για την δημιουργία, ενημέρωση και διαγραφή των δεδομένων, όπως αυτά φαίνονται πιο κάτω.

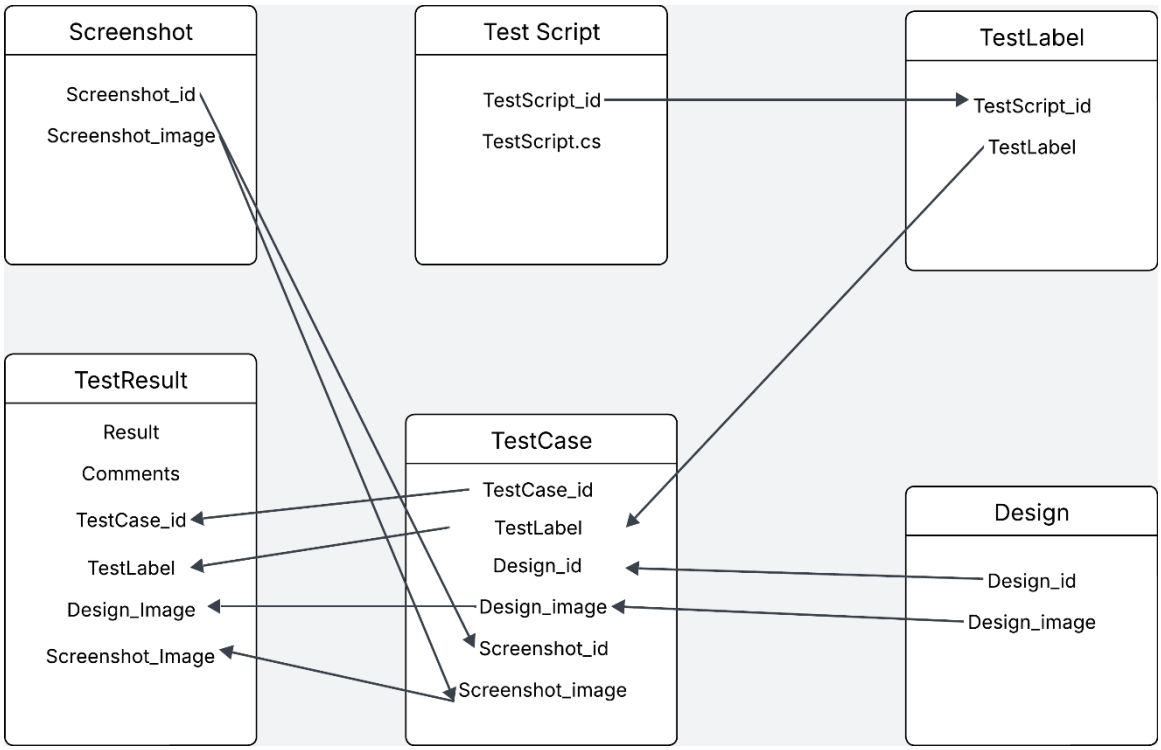
| Σύμβολο | Σημασία |
|---|---|
| **Ορθογώνιο** | **Εξωτερική Οντότητα** |
| **Κύκλος** | **Διαδικασία** |
| **Πορτοκαλί Ορθογώνιο** | **Αποθήκη Δεδομένων** |
| **Βελάκι** | **Ροή δεδομένων** |




***Εικόνα 2 Διάγραμμα Ροής***

### *3.4.3 Λογικό Μοντέλο Δεδομένων / Λεξικό Δεδομένων*

Εδώ παρουσιάζονται τα κύρια δεδομένα που χρησιμοποιούνται στην εφαρμογή και πως σχετίζονται μεταξύ τους. Δεν έχει να κάνει με τις διαδικασίες, αλλά με τις δομές δεδομένων και πως τυγχάνουν χρήσης από την εφαρμογή.




*Εικόνα 3 Διάγραμμα Δεδομένων*

| Όνομα Πεδίου | Περιγραφή | Τύπος Δεδομένων | Πηγή | Χρήσεις |
|---|---|---|---|---|
| Screenshot_id (Screenshot) | Μοναδική ταυτοποίηση στιγμιότυπου | Ακέραιος | Αποθήκευση στιγμιότυπων | Εγγραφή και εκτέλεση δοκιμών, παραγωγή αποτελεσμάτων, διαχείριση δοκιμών |
| Screenshot_Image | Στιγμιότυπο οθόνης | Εικόνα (.png) | Αποθήκευση στιγμιότυπων | Εγγραφή και εκτέλεση δοκιμών, παραγωγή αποτελεσμάτω |

| | | | | |
|---|---|---|---|---|
| | | | | ν, διαχείριση δοκιμών |
| TestScript_id | Μοναδική ταυτοποίηση κωδίκων δοκιμών | Ακέραιος | Αποθήκευση δοκιμών / εγγραφή δοκιμών | Εγγραφή και εκτέλεση δοκιμών, παραγωγή αποτελεσμάτων, διαχείριση δοκιμών |
| TestScript.cs | Αρχείο πηγαίου κώδικα κάθε δοκιμής που έχει εγγραφθεί | Πηγαίος κώδικας .cs | Αποθήκευση δοκιμών / εγγραφή δοκιμών | Εγγραφή και εκτέλεση δοκιμών, παραγωγή αποτελεσμάτων, διαχείριση δοκιμών |
| TestLabel | Η ονομασία (ετικέτα) του κάθε σεναρίου | Κείμενο (string) | Διαχείριση Δοκιμών | Εγγραφή και εκτέλεση δοκιμών, παραγωγή αποτελεσμάτων, διαχείριση δοκιμών |
| Result | Το αποτέλεσμα (pass/fail) της κάθε δοκιμής | Boolean | Αποτέλεσμα δοκιμών | Αποτελέσματα δοκιμών |
| Comments | Τα οπτικά σφάλματα που έχει εντοπίσει η εφαρμογή | Κείμενο (String) | Αποτέλεσμα δοκιμών | Αποτέλεσμα δοκιμών |

## 3.5 Λειτουργικές Απαιτήσεις

Σε αυτό το σημείο ορίζονται οι λειτουργικές απαιτήσεις της εφαρμογής, καταγράφοντας τις συγκεκριμένες συμπεριφορές, τα χαρακτηριστικά και τις λειτουργίες που η εφαρμογή πρέπει να υποστηρίζει. Αυτές οι απαιτήσεις είναι οργανωμένες σε λογικές ομάδες, και η κάθε μια αντιστοιχεί σε ένα κύριο μέρος ή σε μία δυνατότητα του συστήματος. Οι λειτουργικές απαιτήσεις προκύπτουν από τα σενάρια και τις απαιτήσεις χρήστη όπως περιεγράφηκαν στις προηγούμενες περιοχές του εγγράφου αυτού. Η κάθε απαίτηση ταυτοποιείται χρησιμοποιώντας έναν μοναδικό αριθμό ταυτοποίησης και είναι δομημένη ιεραρχικά για να γίνεται πιο σαφής κατανόηση.

### *3.4.1 Λειτουργικές απαιτήσεις: Ομάδα 1*

Ομάδα 1: Εγγραφή σεναρίων δοκιμών

| **Τμήμα/ Ταυτότητα Απαίτησης** | **Ορισμός Απαίτησης** |
|---|---|
| ΛΑ1.0. | Η εφαρμογή θα προσφέρει δυνατότητα καταγραφής δοκιμών ελέγχου |
| ΛΑ1.0.1 | Η εφαρμογή θα προσφέρει γραφικό περιβάλλον για την έναρξη καταγραφής |
| ΛΑ1.0.2 | Η εφαρμογή θα προσφέρει δυνατότητα εκκίνησης των εγγραφών από τον χρήστη |
| ΛΑ1.0.3 | Η εφαρμογή θα διαγράφει αυτόματα τα παλιά σενάρια δοκιμών με την έναρξη της καταγραφής των καινούριων εγγράφων |
| ΛΑ 1.1 | Η εφαρμογή θα πρέπει να αποθηκεύει τα εγγεγραμμένα σενάρια δοκιμών σε αρχεία .cs και σε πλαίσιο ανάπτυξης δοκιμών nUnit |
| ΛΑ 1.1.2 | Η εφαρμογή θα πρέπει αυτόματα να δίνει μοναδική ταυτότητα σε κάθε σενάριο δοκιμής μέσω ονοματοδοσίας των αρχείων |
| ΛΑ 1.1.3 | Τα αρχεία σεναρίων δοκιμών θα πρέπει να είναι διαθέσιμα μέσω των αρχείων συστήματος της εφαρμογής |

### *3.4.2 Λειτουργικές απαιτήσεις: Ομάδα 2*

**Ομάδα 2: Διαχείριση σεναρίων δοκιμών**

| Τμήμα/ Ταυτότητα Απαίτησης | Ορισμός Απαίτησης |
|---|---|
| ΛΑ2.0. | Η εφαρμογή θα προσφέρει δυνατότητα διαχείρισης των εγγεγραμμένων δοκιμών σεναρίων |
| ΛΑ2.0.1 | Θα πρέπει στην διαχείριση σεναρίων δοκιμών σεναρίων να εμφανίζεται η ονοματοδοσία από τον χρήστη μαζί με την ονοματοδοσία που δόθηκε από την εφαρμογή (μοναδική ταυτότητα) |
| ΛΑ2.0.2 | Θα πρέπει να υπάρχει πεδίο εισόδου για την αλλαγή ονομασίας (label) των σεναρίων δοκιμών μέσω του πληκτρολογίου |
| ΛΑ2.0.3 | Θα πρέπει τα εισαγόμενα πεδία εισόδου να αποθηκεύονται σε κατάλληλο αρχείο και να ενημερώνονται σε πραγματικό χρόνο |
| ΛΑ 2.1 | Θα πρέπει να εμφανίζονται τα στιγμιότυπα οθόνης δίπλα από τα σχέδια βάσης στην οθόνη διαχείρισης σεναρίων δοκιμών |
| ΛΑ 2.1.1 | Θα πρέπει στην οθόνη διαχείρισης σεναρίων δοκιμών να υπάρχει πεδίο εισαγωγής εικόνων, που υποστηρίζουν την επιλογή από αρχεία ή την ικανότητα drag-and-drop για την αλλαγή σε πραγματικό χρόνο των σχεδίων βάσης |
| ΛΑ 2.1.2 | Θα πρέπει εάν δεν υπάρχει διαθέσιμη εικόνα βάσης σχεδίου για κάποιο σενάριο δοκιμής, να εμφανίζεται αντίστοιχο μήνυμα για πρόσθεση εικόνας σχεδίου βάσης |

### *3.4.3 Λειτουργικές απαιτήσεις: Ομάδα 3*

**Ομάδα 3: Εκτέλεση σεναρίων δοκιμών**

| Τμήμα/ Ταυτότητα Απαίτησης | Ορισμός Απαίτησης |
|---|---|
| ΛΑ3.0. | Θα πρέπει να υπάρχει διασύνδεση με μοντέλο Chat-GPT 4o σε Azure OpenAI Service, για την εκτέλεση των δοκιμών |

| Τμήμα/ Ταυτότητα Απαίτησης | Ορισμός Απαίτησης |
|---|---|
| ΛΑ3.0.1 | Θα πρέπει μέσω γραφικής διεπαφής χρήστη να αποστέλλεται αίτημα εκτέλεσης δοκιμών ελέγχου |
| ΛΑ3.0.2 | Θα πρέπει να υπάρχει μπάρα αναμονής μέχρι την εξαγωγή των αποτελεσμάτων |
| ΛΑ3.1. | Θα πρέπει να παράγεται αρχείο αναφοράς των αποτελεσμάτων δοκιμών |
| ΛΑ3.1.1 | Θα πρέπει το αρχείο αναφοράς να είναι αρχείο HTML, που εμφανίζει τον αρ. Δοκιμών, αρ. Επιτυχημένων δοκιμών, αρ. Αποτυχημένων δοκιμών και το αποτέλεσμα κάθε δοκιμής μαζί με την απάντηση που δόθηκε από το μοντέλο Chat-GPT 4o |
| ΛΑ 3.1.2 | Θα πρέπει η παραγωγή αναφοράς HTML να γίνεται απευθείας μετά την εκτέλεση των σεναρίων δοκιμής και να αποθηκεύεται στα αρχεία του πηγαίου κώδικα της εφαρμογής |
| ΛΑ 3.1.3 | Θα πρέπει το αρχείο αναφοράς να ανοιγεί αυτόματα με το πέρας της εκτέλεσης των δοκιμών |

## 3.6 Μη-Λειτουργικές Απαιτήσεις

Εδώ γίνεται σύντομη περιγραφή των μη-λειτουργικών απαιτήσεων της εφαρμογής αυτοματοποιημένων δοκιμών. Αντιθέτως με τις λειτουργικές απαιτήσεις, εδώ θα αναφερθούν συντόμως οι απαιτήσεις για την απόδοση της εφαρμογής, τους περιορισμούς της και υπό ποιες συνθήκες. Αυτοί οι περιορισμοί φροντίζουν η εφαρμογή να είναι αξιόπιστη, αποτελεσματική, επεκτάσιμη, ασφαλής και εύχρηστη.

### *3.6.1 Απαιτήσεις διεπαφής*

#### 3.6.1.1 Διεπαφές Υλικού

Καθώς η εφαρμογή θα πρέπει να είναι εκτελέσιμη σε συσκευές γενικές χρήσης, δεν θα πρέπει να έχει πολύ υψηλές απαιτήσεις υλικού. Θα πρέπει να υποστηρίζει σύνδεση στο διαδίκτυο. Οι ελάχιστες απαιτήσεις υλικού θα πρέπει να έχουν ως εξής:

- CPU: Τουλάχιστον τεταρτοπύρηνος (για την παράλληλη εγγραφή δοκιμών και την εκτέλεση της εφαρμογής και των οδηγό φυλλομετρητή)
- Κεντρική Μνήμη: Τουλάχιστον 4 GB
- Αποθηκευτικός χώρος: Τουλάχιστον 10 GB για την εγκατάσταση του Visual Studio, του κώδικα της εφαρμογής, των σχετικών βιβλιοθηκών και την αποθήκευση των εικόνων και αναφορών.
- GPU: Δεν απαιτείται

#### 3.6.1.2 Διεπαφές Λογισμικού

Θα πρέπει να υπάρχουν διασυνδέσεις με τα εξής λογισμικά:

- Διεπαφή με τον οδηγό φυλλομετρητή Playwright:
    - Χρησιμοποιείται στην εγγραφή των δοκιμών ελέγχου μέσω του Playwright Codegen
    - Συνδέεται με την εφαρμογή μέσω της βιβλιοθήκης του Playwright στη C#
    - Χρησιμοποιείται στην καταγραφή των στιγμιότυπων οθόνης
- Πλαίσιο δοκιμών nUnit
    - Παραγωγή κώδικα σε nUnit μέσω του C#
    - Εκτέλεση των δοκιμών σε nUnit
- Σύστημα αρχείων Windows
    - Καθώς πρόκειται για εφαρμογή που τρέχει σε Windows, πρέπει να υπάρχει διασύνδεση με την διαχείριση αρχείων των Windows
    - Χρησιμοποιείται στην διαγραφή, ανάκληση και αποθήκευση των απαραίτητων αρχείων της εφαρμογής
- Εντολές συστήματος Windows
    - Για την σωστή διαχείριση των αρχείων του Windows
    - Για την σωστή ταυτοχρονία της εκτέλεσης εφαρμογής, του Playwright Codegen, και τις σωστής πρόσβασης αυτών στα κοινά assets
- C# .NET και στις σχετικές βιβλιοθήκες

- Για την εκτέλεση του πηγαίου κώδικα

#### 3.6.1.3 Διεπαφές Επικοινωνίας

- HTTP/HTTPS REST APIs

  Γίνεται η χρήση του API του πράκτορα Chat-GPT 4o όπου μέσω των σωστών μυστικών κωδικών, αποστέλλονται τα αιτήματα σύγκρισης

- TCP/IP Δίκτυο

  Για την πρόσβαση στο υπό εξέταση σύστημα διαδικτύου

#### 3.6.1.4 Απαιτήσεις Δεδομένων

Η εφαρμογή δεν απαιτεί μετάπτωση δεδομένων από κάποια προηγουμένη πλατφόρμα. Όμως, κατά την διάρκεια των εγγραφών και της εκτέλεσης των δοκιμών, απαιτούνται συγκεκριμένες μορφές δεδομένων:

- Τύποι εικόνας: Κατά κύριο λόγο, η απαιτούμενη μορφή εικόνων σχεδίων βάσης και στιγμιότυπων οθόνης, θα πρέπει να είναι .png
- Τύπος εγγραφών: Θα πρέπει να υπάρχει υποστήριξη JSON αρχείων για την αποθήκευση σχετικών δεδομένων με τις δοκιμές για καλύτερη διαχείριση, και υποστήριξη HTML για τις αναφορές

## 3.7 Επιχειρησιακές απαιτήσεις

Αυτό το τμήμα περιγράφει τις επιχειρησιακές απαιτήσεις για την εφαρμογή των αυτοματοποιημένων οπτικών δοκιμών. Αυτές οι απαιτήσεις ορίζουν τις συνθήκες κάτω από τις οποίες το σύστημα πρέπει να λειτουργεί ούτως ώστε να υπάρχει σταθερότητα, αξιοπιστία και διαθεσιμότητα. Επικεντρώνεται στο πως η εφαρμογή αναμένεται να λειτουργεί υπό πραγματικές συνθήκες συμπεριλαμβανομένων των πτυχών της ασφαλείας, παρακολούθησης, λειτουργίας και συμβατότητας με τις υποδομές ανάπτυξες. Αυτές οι απαιτήσεις καθορίζουν το πως θα συντηρείται και θα παρακολουθείται η εφαρμογή καθόλη την διάρκεια του κύκλου ανάπτυξής της, ευθυγραμμίζοντάς την με τις οργανωτικές ανάγκες και τα επιχειρησιακά πρότυπα.

### *3.7.1 Ασφάλεια και Ιδιωτικότητα*

Α. Εάν υπάρξουν παραβιάσεις των μέτρων ασφαλείας είναι πιθανόν να υπάρξουν οι εξής συνέπειες

1. Καταστροφή η διαφθορά δεδομένων
2. Διαρροή ιδιωτικών σχεδίων ή κώδικα
3. Διαρροή των κωδικών για πρόσβαση στον πράκτορα Chat GPT-4o
4. Φθορά λογισμικού

Β. Η ασφάλεια που απαιτείται για την εφαρμογή

1. Φυσική ασφάλεια
2. Πρόσβαση μόνο από τον χρήστη με τα κατάλληλα προνόμια
3. Πρόσβαση στον κώδικα μόνο για τους ενδιαφερόμενους μέσω του Azure DevOps
4. Χρησιμοποίηση κωδικών πρόσβασης μέσω συσκευής στο VPN

### *3.7.2 Αξιοπιστία*

Α.

- Ζημίες που μπορεί να προκύψουν στην αποτυχία της εφαρμογής
    a) Απώλεια εγγράφων δοκιμών ελέγχου
    b) Απώλεια εγγράφων αποτελεσμάτων δοκιμών
    c) Απώλεια παραγωγικότητας των ομάδων δοκιμών

1. What is the minimum acceptable level of reliability?

Β. Απαιτούμενη αξιοπιστία:

1. Μέσος Χρόνος μεταξύ Αποτυχίας: 1 μήνας
2. Μέσος Χρόνος επιδιόρθωσης: 15 λεπτά – 2 ώρες

### *3.7.3 Ανάκτηση*

Α. Εάν η εφαρμογή δεν είναι διαθέσιμη στους χρήστες λόγω αποτυχίας συστήματος, θα πρέπει να είναι ανακτήσιμη εντός δύο ωρών μέσω του αποθετηρίου κώδικα

Β. Στην περίπτωση που δεν είναι διαθέσιμο το αποθετήριο τότε ο χρόνος ανάκτησης δεν μπορεί να προβλεπτεί καθώς αυτό εξαρτάται από την υποδομή του νέφους

### *3.7.4 Διαθεσιμότητα Συστήματος*

Το σύστημα υπό κανονικές συνθήκες, θα πρέπει να είναι διαθέσιμο ανά πάσα στιγμή, καθώς δεν υπάρχει ωράριο λειτουργίας του πράκτορα Chat-GPT 4o. Όμως, εάν αυτό

αποτύχει, τότε αναμένεται να μην μπορεί να επιδιορθωθεί κατά τις μη εργάσιμες μέρες και ώρες (δηλ. τα σαββατοκύριακα και ώρες εκτός 0800-1800).

### *3.7.5 Γενική Απόδοση*

A. Ώρα απόκρισης

   a. Δεν θα πρέπει να χρειάζεται πάνω από μισό λεπτό για την αποθήκευση κάθε σεναρίου δοκιμής

   b. Δεν θα πρέπει να χρειάζεται πάνω από 2 λεπτά για την απόκριση του πράκτορα Chat GPT 4o για κάθε σενάριο δοκιμής

   c. Δεν θα πρέπει να χρειάζεται πάνω από 10 λεπτά για την παραγωγή της αναφοράς HTML ανά 100 δοκιμές

   d. Δεν θα πρέπει να απαιτούνται πάνω από 6000 Chat GPT 4o input tokens και 1200 output tokens ανά αίτημα για την εκτέλεση ενός σεναρίου δοκιμής

B. Ρυθμός διεκπεραίωσης

   a. Συνυπολογίζοντας τα παραπάνω, θα πρέπει να υπάρχει περίπου ελάχιστος ρυθμός διεκπεραίωσης 24 δοκιμών την ώρα μαζί με την εγγραφή, και 50 δοκιμών την ώρα για εκτέλεση.

C. Ο ρυθμός χρήσης από χρήστη την ημέρα, αναμένεται γύρω στις 50-100 εγγραφές την ημέρα, και 200-300 εκτέλεσης την ημέρα.

### *3.7.6 Χωρητικότητα*

Θα πρέπει να υπάρχει η δυνατότητα μέγιστης αποθήκευσης γύρω στα 300 σενάρια δοκιμών. Υποθέτουμε πως περίπου οι εικόνες στιγμιότυπου και σχεδίων βάσης των σεναρίων θα είναι στο 1 MB και του αρχείου κώδικα για το σενάριο γύρω στο 1KB, τότε θα χρειάζονται περίπου 300MB για σκοπούς αποθήκευσης.

### *3.7.7 Ανάκληση Δεδομένων*

Το σύστημα θα πρέπει να έχει αποθηκευμένα τα εγγεγραμμένα σενάρια δοκιμών μόνο μέχρι την επόμενη εγγραφή δοκιμών. Εάν δεν γίνει καινούρια εγγραφή δοκιμών, τότε θα πρέπει να παραμένουν αποθηκευμένα επ’ αόριστόν. Ομοίως, οι εικόνες σχεδίων βάσης,

εάν δεν αντικατασταθούν θα πρέπει να παραμείνουν αποθηκευμένες. Τα αποτελέσματα επίσης θα πρέπει να μείνουν αποθηκευμένα μέχρι την επόμενη εκτέλεση.

### *3.7.8 Χειρισμός Σφαλμάτων*

Τα σφάλματα θα πρέπει να τυγχάνουν διαχειρίσεως από την κονσόλα λειτουργικού συστήματος Windows για το όταν τυγχάνουν σε αυτό το περιβάλλον και αντίστοιχα από την γραφική διεπαφή χρήστη όταν αφορούν σε σφάλματα αυτής της περιοχής. Δεν είναι εγγυημένο ότι η ροή του προγράμματος θα είναι κανονική παρά τα σφάλματα, εξού θα απαιτείται η σωστή χρήση της εφαρμογής

### *3.7.9 Κανόνες Επικυροποίησης*

Εδώ καταγράφονται όλες οι επικυροποίησης για τα δεδομένα που πρέπει να γίνονται για την σωστή εκτέλεση της εφαρμογής

- Εικόνες σχεδίων και στιγμιότυπων: Θα πρέπει να είναι .png η .jpeg για συμβατικότητα
- Αρχεία Δοκιμών Ελέγχου: Θα πρέπει να είναι .cs με nUnit και σε Playwright
- Αιτήματα με το μοντέλο Chat GPT – 4o: Θα πρέπει να είναι δομημένα σε μορφή HTTP – Rest API

### *3.7.10 Συμβάσεις*

- Ο πηγαίος κώδικας θα πρέπει να ακολουθεί όλες τις συμβάσεις του κώδικα C# για καλύτερη συντήρηση και επεκτασιμότητα
- Θα πρέπει να ακολουθείται η πρέπουσα δομή για τα αιτήματα των HTTP – REST API αιτημάτων
- Θα πρέπει η αναφορά που παράγεται για τα αποτελέσματα δοκιμών να είναι γραμμένη σε HTML πρότυπα που είναι εκτελέσιμα από σύγχρονους φυλλομετρητές

## 3.8. Σχεδίαση

Σε αυτό το σημείο γίνεται αναφορά για τις προσδοκίες ως προς την σχεδίαση του συστήματος. Εδώ αναλύονται τα στοιχεία που θα πρέπει να υπάρχουν σε κάθε οθόνη για την σωστή σχεδίαση των ροών της εφαρμογής. Όλα τα στοιχεία θα πρέπει να είναι δομημένα με βιβλιοθήκη στη C# για γραφικά στοιχεία των Windows.

| Στοιχείο | Περιγραφή |
|---|---|
| 1.Οθόνη Εκκίνησης | |
| 1.1 Λογότυπο Techlink | Θα πρέπει να υπάρχει το λογότυπο της εταιρείας στην κύρια οθόνη |
| 1.2 Κουμπιά επιλογών ροών (καταγραφή, εκτέλεση, διαχείριση δοκιμών) | Θα πρέπει να είναι σχεδιασμένα ως κουμπιά γραφικών Windows |
| 2. Έναρξη δοκιμών | |
| 2.1 Παράθυρο ενημέρωσης του χρήστη ότι η εγγραφή των δοκιμών εκτελείται αυτή τη στιγμή | Θα πρέπει να εμφανίζεται παράθυρο τύπου pop-up το οποίο παραμένει ενεργό καθώς γίνεται καταγραφή των δοκιμών |
| 2.2 Περιβάλλον εγγραφής Playwright Codegen | Θα πρέπει να γίνεται εκκίνηση του περιβάλλοντος εγγραφής Playwright Codegen |
| 2.3 Εισαγωγή αριθμού δοκιμών για εγγραφή | Πριν από την εκκίνηση εγγραφής θα πρέπει να εμφανίζεται πεδίο εισαγωγής με κουμπί υποβολής για τον αρ. Εγγραφών που θα γίνουν |
| 2.4 Ενημέρωση χρήστη για την επιτυχημένη αποθήκευση | Αφού ο χρήστης γράψει τον αρ. Εγγραφών θα πρέπει να εμφανίζεται pop-up window ότι ήταν επιτυχής η εγγραφή των δοκιμών |
| 3. Διαχείριση Δοκιμών | |
| 3.1 Πίνακας δοκιμών | Θα πρέπει να εμφανίζεται καλά στοιχισμένος πίνακας με την κάθε δοκιμή |
| 3.2 Ονόματα δοκιμών | Θα πρέπει να υπάρχει πεδίο εισόδου |
| 3.3 Φωτογραφίες Σχεδίων | Θα πρέπει να υπάρχει κουτί εισαγωγής φωτογραφίας σχεδίου βάσης εάν δεν έχει καταχωρηθεί κάποια σε κάθε δοκιμή. Θα πρέπει να υποστηρίζει και το drag and drop και την αναζήτηση. Επίσης, δίπλα από κάθε φωτογραφία θα πρέπει να υπάρχει διαγραφή της φωτογραφίας |

| | |
|---|---|
| 3.4 Πλοήγηση Πίνακα | Εάν υπάρχουν πολλές δοκιμές θα πρέπει να υπάρχει επιλογή scrolling για την πλοήγηση σε όλον τον πίνακα. |
| 4. Εκτέλεση Δοκιμών | |
| 4.1 Μπάρα αναμονής | Με την έναρξη της εκτέλεσης θα πρέπει να εμφανίζεται μπάρα αναμονής για να ενημερώνεται ο χρήστης για την σωστή εκτέλεση |
| 4.2 Ανακοίνωση αποτελεσμάτων | Θα πρέπει να εμφανίζεται pop up window με το πέρας της εκτέλεσης για να ενημερωθεί ο χρήστης πως έχουν αποθηκευτεί τα αποτελέσματα |
| 4.3 Αναφορά αποτελεσμάτων | Θα πρέπει να παράγεται αναφορά τύπου HTML, με πίνακα με τα στοιχεία των δοκιμών μαζί με τα μηνύματα αποτελεσμάτων. Σε αυτό θα πρέπει επίσης να καταγράφονται πόσα ήταν επιτυχημένα και πόσα αποτυχημένα. |

# Κεφάλαιο 4



## Κεφάλαιο 4: Υλοποίηση

Η υλοποίηση της εφαρμογής πραγματοποιήθηκε με στόχο την δημιουργία ενός λειτουργικού εργαλείου που να υποστηρίζει την ημι-αυτόνομη ανίχνευση οπτικών σφαλμάτων και να πληροί τις απαιτήσεις που περιεγράφηκαν στο κεφάλαιο 3. Το εργαλείο ενσωματώνει μία μέθοδο καταγραφής σεναρίων, μία μέθοδο αντιστοίχισης με σχέδια βάσης (π.χ. Figma σε μορφή εικόνων), αποστολής ερωτήματος στο μοντέλο GPT-4o και παραγωγής αναφοράς αποτελεσμάτων. Σε αυτό το κεφάλαιο θα γίνει μια επεξήγηση του τρόπου υλοποίησης της εφαρμογής καθώς και γενικότερη περιγραφή των βημάτων αυτής καθόλη την διάρκεια ανάπτυξης της εφαρμογής. Θα εξετάσουμε την υλοποίηση από τρεις κύριες απόψεις 1) αρχιτεκτονική 2) τεχνικές εξαρτήσεις 3) υλοποίηση των λειτουργιών καθ' εαυτών και εν συνεχεία, θα δούμε τι δυνατότητες επέκτασης θα μπορούσαν να γίνουν στην εφαρμογή με βάση την δομή της.

### 4.1 Αρχιτεκτονική Εφαρμογής

Η εφαρμογή έχει μια αρχιτεκτονική τριών επιπέδων, με το κάθε επίπεδο να ανταποκρίνεται σε λειτουργίες που είναι απαραίτητες για την ορθή εκτέλεση της εφαρμογής. Τα επίπεδα έχουν ως εξής:

1. Επίπεδο Παρουσίας: Γραφική Διεπαφή Windows Forms και αναφορά HTML

2. Επίπεδο Επιχειρησιακής Λογικής: Διαχείριση σεναρίων, συσχέτιση εικόνων, κλήση API
3. Επίπεδο Πρόσβασης Δεδομένων: Διαχείριση αρχείων (.cs, .png, .json), χρήση τοπικού συστήματος αρχείων.

Το κάθε επίπεδο είναι απομονωμένο με χρήση ορθών κλάσεων/αντικειμένων σε επίπεδο κώδικα στη C# και υπάρχει κατάλληλη εξάρθρωση (modularity) για ευκολία στην συντήρηση και στην επέκταση.

Η γραφική διεπαφή χρήστη επικοινωνεί με την επιχειρησιακή λογική αποκλειστικά μέσω μεθόδων ούτως ώστε να υπάρχει συμβατότητα και δυνατότητα δοκιμής του. Το επίπεδο λογικής χειρίζεται την υλοποίηση όλης της λειτουργικότητας και προστατευτεί τα δεδομένα από την γραφική διεπαφή χρήστη (φορτώνει μόνο τα απαιτούμενα). Το επίπεδο δεδομένων φέρει ευθύνη για την ανάγνωση/επεξεργασία/διαγραφή των αρχείων όταν αυτό απαιτείται από τις λειτουργείς της εφαρμογής.

### 4.*1.1 Γενική Δομή*

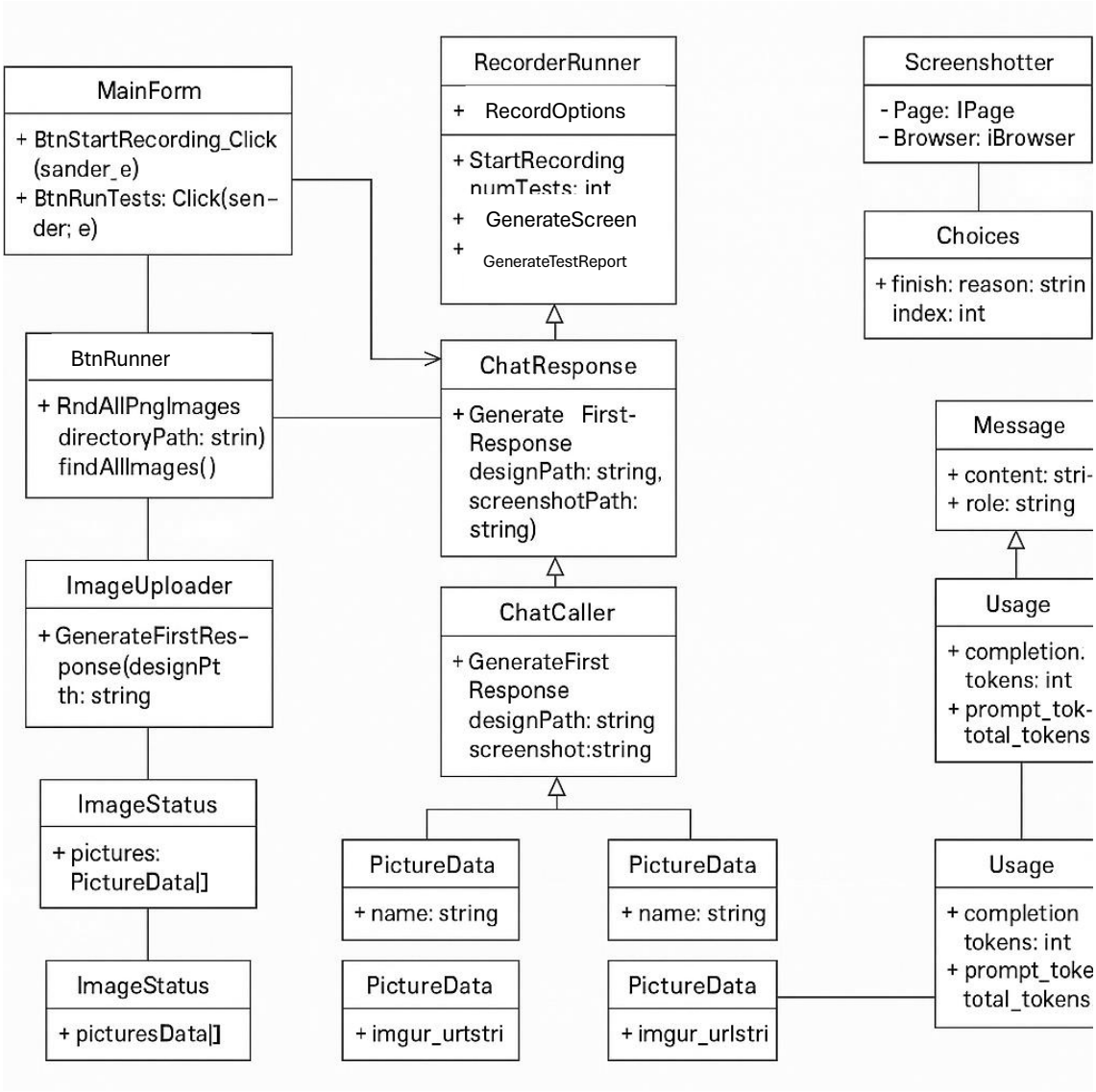
Γενικώς ακολουθούνται οι αρχές SOLID:

- Single Responsibility: Υπάρχει μόνο ένας λόγος αλλαγής μίας κλάσης, δηλαδή ευθύνονται για μια συγκεκριμένη λειτουργία
- Dependency Inversion: Δεν υπάρχει εξάρτηση από κλάσεις χαμηλού επιπέδου σε κλάσεις υψηλού επιπέδου
- Open/Closed: Υπάρχει χώρος επέκτασης για κάθε μονάδα
- Liskov Substitution: Μπορούν να χρησιμοποιηθούν με ασφάλεια οι αρχές κληρονομικότητας
- Interface Segregation: Δεν χρησιμοποιούνται άσχετες διεπαφής με την εκτελούμενη διεργασία

Οι διεργασίες διαχωρίζονται σε κλάσεις ούτως ώστε να διατηρούνται ανεξάρτητες, όπως φαίνεται στο πιο κάτω διάγραμμα.

- RecorderRunner: Η λογική πυρήνα για την ανάγνωση και εκτέλεση των δοκιμών. Ενεργοποιεί το Playwright codegen, αλλάζει τον κώδικα δοκιμών για να βγάζει στιγμιότυπα οθόνης και παράγει αναφορές HTML.

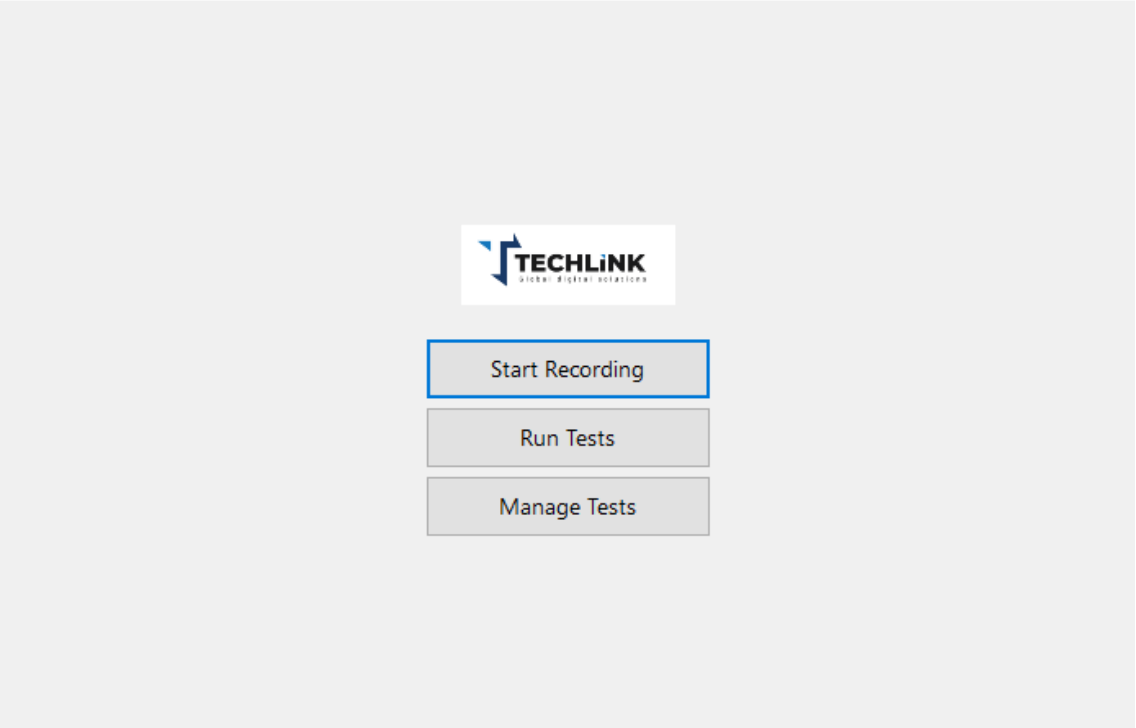
- ChatCaller: Ενεργοποιεί και καλεί το μοντέλο ChatGPT-4o στο OpenAI. Λαμβάνει 2 εικόνες σε base64 και ένα prompt και το στέλνει στο μοντέλο, και λαμβάνει πίσω την απάντηση.
- ChatResponse: Χρησιμοποιείται στο ChatCaller και χρησιμοποιείται για την επεξεργασία της απάντησης, αφού είναι η δομή του.
- ImageUploader: Διαχείριση των εικόνων
- ButtonHandler: Κλάση για τον χειρισμό της διεπαφής χρήστη και των κουμπιών της
- MainForm: Η κύρια φόρμα γραφικής διεπαφής χρήστη και η διαχείρισή της



### *4.1.2 Διεπαφή Χρήστη*

Εδώ εμφανίζεται η τελική διεπαφή χρήστη όπως αυτή υλοποιήθηκε μέσω των Windows Graphic Forms και της αναφοράς HTML. Γίνεται καταγραφή ανά οθόνη.

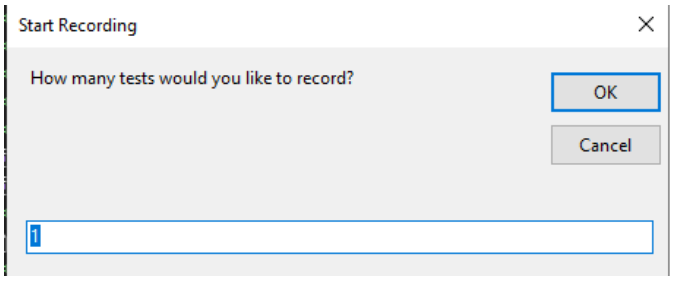
1. Κύρια οθόνη:



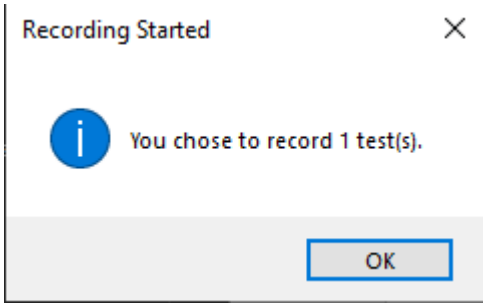

2. Διαχείριση Δοκιμών

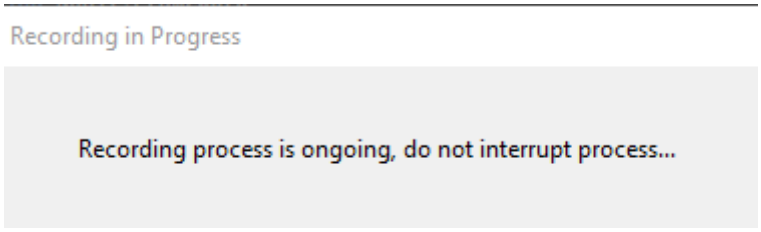

3. Καταγραφή δοκιμών

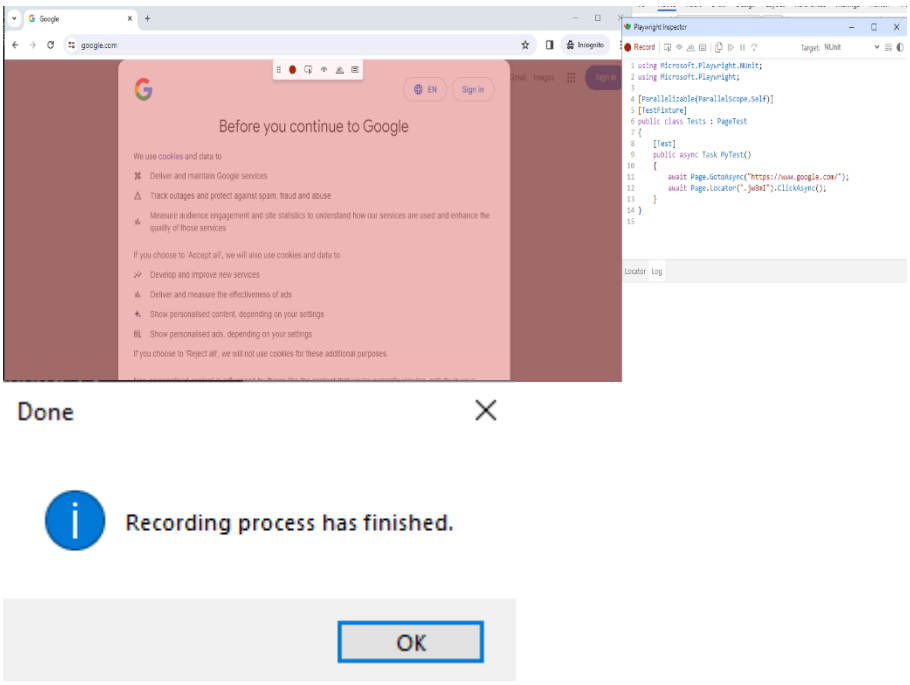
    a. Παράθυρο επιλογής αρ. δοκιμών



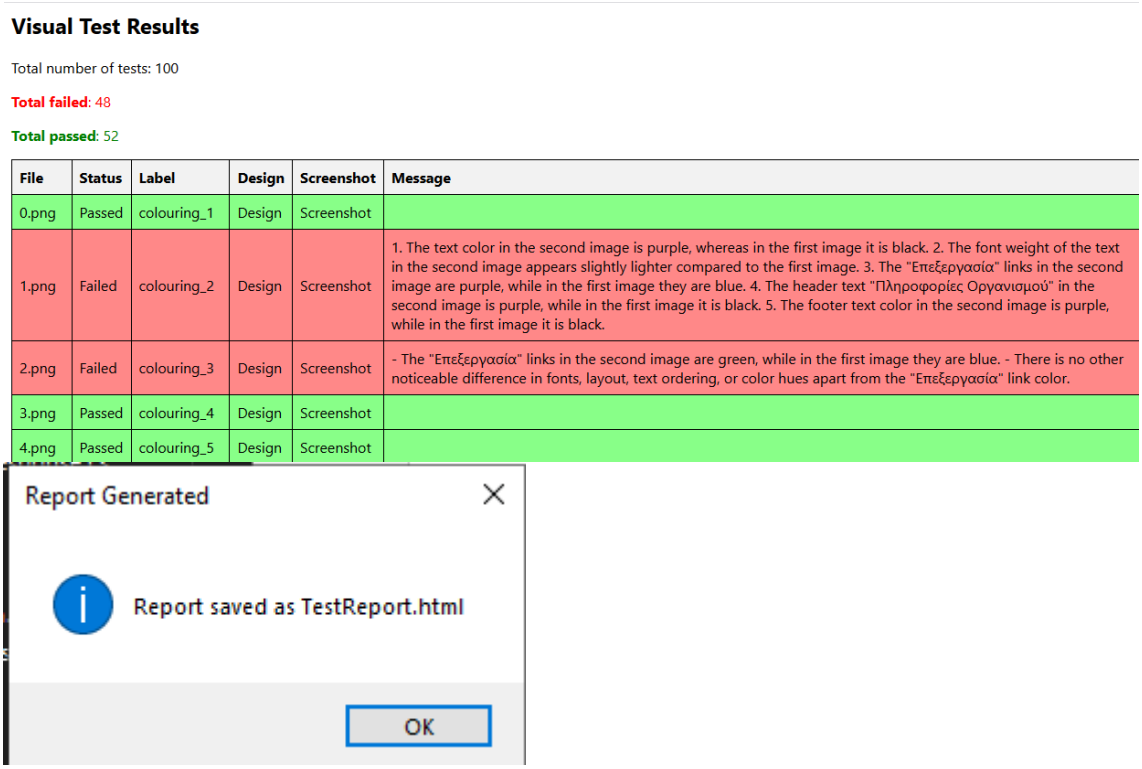

b. Διαδικασία Εγγραφής




4. Εκτέλεση Δοκιμών



**Visual Test Results**

Total number of tests: 100

**Total failed**: 48

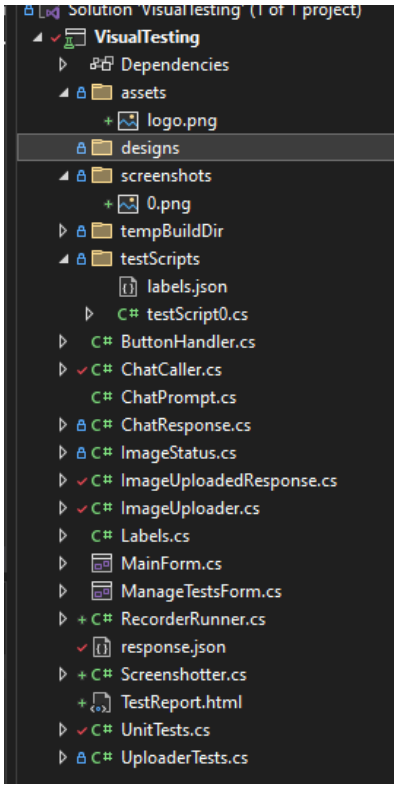
**Total passed**: 52

| File | Status | Label | Design | Screenshot | Message |
|---|---|---|---|---|---|
| 0.png | Passed | colouring_1 | Design | Screenshot | |
| 1.png | Failed | colouring_2 | Design | Screenshot | 1. The text color in the second image is purple, whereas in the first image it is black. 2. The font weight of the text in the second image appears slightly lighter compared to the first image. 3. The "Επεξεργασία" links in the second image are purple, while in the first image they are blue. 4. The header text "Πληροφορίες Οργανισμού" in the second image is purple, while in the first image it is black. 5. The footer text color in the second image is purple, while in the first image it is black. |
| 2.png | Failed | colouring_3 | Design | Screenshot | - The "Επεξεργασία" links in the second image are green, while in the first image they are blue. - There is no other noticeable difference in fonts, layout, text ordering, or color hues apart from the "Επεξεργασία" link color. |
| 3.png | Passed | colouring_4 | Design | Screenshot | |
| 4.png | Passed | colouring_5 | Design | Screenshot | |




### *4.1.3 Δομή Αρχείων*

Τα αρχεία είναι δομημένα και ιεραρχημένα με βάση την ρίζα του έργου. Υπάρχει φάκελος για τα στιγμιότυπα (screenshots), για τα σχέδια βάσης (designs) και ειδικό αρχείο για τα ονόματά τους. Επίσης υπάρχει φάκελος για τα εγγεγραμμένα σενάρια δοκιμών. Ταυτόχρονα, τα αρχεία κώδικα μένουν εντός του ίδιου επιπέδου για κοινή συνεννόηση σχετικών path. Υπάρχουν και άλλοι βοηθητικοί φάκελοι όπως τα assets για χρήσιμα αρχεία όπως π.χ. το λογότυπο και το tempBuildingDir για σκοπούς εκτέλεσης αρχείων στις διάφορες διαδικασίες.




## 4.2 Τεχνικές Εξαρτήσεις

Για την ανάπτυξη της εφαρμογής χρησιμοποιήθηκαν πολλές ήδη υπάρχουσες τεχνολογίες για την ευκολότερη, πιο ασφαλή και αποτελεσματική ανάπτυξη της εφαρμογής. Συνεπώς, για την σωστή εκτέλεση του κώδικα του έργου απαιτούνται τα εξής:

- Microsoft.Playwright: Για τον οδηγό φυλλομετρητή, εγγραφή και εκτέλεση δοκιμών
- NUnit για την εγγραφή και εκτέλεση δοκιμών
- OpenAI API (Chat-GPT 4o): Azure Foundry Deployment
- System.IO: Για την διαχείριση τοπικών αρχείων
- Newtonsoft.Json: Για χειρισμό των μεταδεδομένων και αποτελεσμάτων

Για την συμβατότητα των πακέτων, καλύτερο είναι να ακολουθηθούν οι εξής εκδόσεις, όπως είναι γραμμένες στο αρχείο .sln του έργου:

```
            <PackageReference Include="Microsoft.NET.Test.Sdk"
Version="17.6.0" />
            <PackageReference Include="Microsoft.Playwright"
Version="1.42.0" />
            <PackageReference Include="Microsoft.Playwright.MSTest"
Version="1.42.0" />
            <PackageReference Include="Microsoft.Playwright.NUnit"
Version="1.42.0" />
```

```
            <PackageReference Include="Microsoft.Playwright.TestAdapter"
Version="1.42.0" />
            <PackageReference Include="NUnit" Version="3.13.3" />
            <PackageReference Include="NUnit3TestAdapter" Version="4.2.1"
/>
            <PackageReference Include="NUnit.Analyzers" Version="3.6.1" />
            <PackageReference Include="coverlet.collector" Version="6.0.0"
/>
            <PackageReference Include="System.Management" Version="9.0.3"
/>
      </ItemGroup>
```

## 4.3 Υλοποίηση Βασικών Λειτουργιών

Εδώ γίνεται σύντομη αναφορά στην έννοια πίσω από τον τρόπο με τον οποίο αναπτύχθηκαν οι βασικές λειτουργίες. Καθώς ο κώδικας είναι προστατευμένος από πνευματικά δικαιώματα, γίνεται μόνο μια βασική καταγραφή των λειτουργικών template που χρησιμοποιήθηκαν στην ανάπτυξη.

### *4.3.1 Εγγραφή Δοκιμών*

Η εγγραφή υλοποιείται ως wrapper του Playwright Codegen μέσω εντολών του περιβάλλοντος εντολών του windows και εκτελούνται στη C#, με επιλογές αποθήκευσης σε αρχείο.

```
Process.Start(new ProcessStartInfo {
    FileName = "cmd.exe",
    Arguments = "/C playwright codegen https://mytestpage.com",
    UseShellExecute = false
});
```

Η εγγραφή αποθηκεύει αρχεία .cs με τις εντολές πλοήγησης. Κάθε αρχείο συσχετίζεται με μοναδικό ID. Μέσω παρόμοιας διαδικασίας command line γίνεται και η εκτέλεση των δοκιμών για την αποθήκευση των screenshot.

### *4.3.2 Συσχέτιση Σχεδίων*

Ο χρήστης μπορεί να φορτώσει μέσω του GUI τις εικόνες σχεδίου βάσης και να τα συσχετίσει με screenshots δοκιμών όπως και επίσης ονόματα.

```
{
```

```
  "test_id": "123",
  "screenshot": "screenshot_123.png",
  "design": "design_123.png",
  "label" : "layout_test"
}
```

### *4.3.3 Εκτέλεση Ελέγχων*

Η εκτέλεση γίνεται με ανάγνωση του JSON, σύγκριση εικόνων και αποστολή μέσω API call:

```
var content = new MultipartFormDataContent();
content.Add(new
ByteArrayContent(File.ReadAllBytes("screenshot.png")),
"image1");
content.Add(new
ByteArrayContent(File.ReadAllBytes("design.png")), "image2");
content.Add(new StringContent(prompt), "prompt");

var result = await client.PostAsync(apiUrl, content);
```

Δομή αιτήματος:

```
string instruction = "Find differences. Do not focus much on
text contents, focus on design differences. If there are
differences that are errors, in the first line write True,
otherwise False. After that line, list the errors. Only state
differences in fonts for the text. Try examining small
differences in color hues, notice the layout in text contents
and ordering, and also small design differences.";
 // Initialize HTTP client
 using (var _client = new HttpClient())
 {
     // Add the API key to the request header
     _client.DefaultRequestHeaders.Add("api-key", apiKey);

     // Create the payload
     var payload = new
     {
         messages = new object[]
         {
             new
             {
                 role = "system",
                 content = new object[]
         {
             new
             {
                 type = "text",
                 text = instruction
             }
         }
             },
             new
             {
```

```
                role = "user",
                content = new object[]
        {
            new
            {
                type = "image_url",
                image_url = new
                {
                    url =
$"{designBase64}"
                }
            },
            new
            {
                type = "image_url",
                image_url = new
                {
                    url =
$"{screenBase64}"
                }
            },
        }
            }
        },
        temperature = 0.3,
        top_p = 0.2,
        max_tokens = 800,
        stream = false
    };
```

**// Send the POST request**

```
    var response = await _client.PostAsync(
        endpointUrl,
        new StringContent(JsonConvert.SerializeObject(payload),
Encoding.UTF8, "application/json")
    );

    // Handle the response
    if (response.IsSuccessStatusCode)
    {
        var responseData =
JsonConvert.DeserializeObject<dynamic>(await
response.Content.ReadAsStringAsync());
        string generatedResponse =
responseData?.choices?[0]?.message?.content?.ToString() ?? "No
response received";
        return generatedResponse;
    }
    else
    {
        return $"Error: {response.StatusCode},
{response.ReasonPhrase}";
    }
```

Το αποτέλεσμα ερμηνεύεται ως pass/fail με βάση το πρώτο token στην απάντηση (“true”/”false”).

### *4.3.4 Παραγωγή Αναφοράς*

Με βάση τα παραγόμενα αποτελέσματα από το 4.4.3, μπορεί να γίνει η παραγωγή αναφοράς με βάση το HTML template, για την παρουσίαση των αποτελεσμάτων, που περιλαμβάνει κωδικοποίηση χρώματος για το αποτέλεσμα (κόκκινο – αποτυχία, πράσινο – επιτυχία), μηνύματα σφαλμάτων, τις ονομασίες των δοκιμών και την αρίθμηση συνολικών δοκιμών και συνολικών αποτυχημένων/επιτυχημένων.

```
<html>
<head>
<title>Visual Test Report</title>
<style>
body { font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif; }
h2 { font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif; }
table { width: 100%; border-collapse: collapse; }
th, td { text-align: left; padding: 8px; }
th { background-color: #f2f2f2; }
</style>
</head>
<body>
<h2>Visual Test Results</h2>
<p>Total number of  tests: 100</p>
<p style='color: red'><strong>Total failed</strong>: 41</p>
<p style='color: green'><strong>Total passed</strong>: 59</p>
<table border='1' cellspacing='0' cellpadding='5'>
<tr><th>File</th><th>Status</th><th>Label</th><th>Design</th><th>Scree
nshot</th><th>Message</th></tr>
<tr style='background-color:#8f8'>
<td>0.png</td>
<td>Passed</td>
<td>colouring_1</td>
<td><img src='designs\0.png' alt='Design' style='max-width:200px; max-
height:200px;'></td>
<td><img src='screenshots\0.png' alt='Screenshot' style='max-
width:200px; max-height:200px;'></td>
<td></td>
</tr>
```

## 4.4 Δυνατότητες Επέκτασης

Η παρούσα υλοποίηση ακολουθεί αρχές αρθρωτικής σχεδίασης και βασίζεται σε ανεξάρτητα, ελαφρώς συνδεδεμένα υποσυστήματα. Αυτό επιτρέπει σημαντική επεκτασιμότητα, καθιστώντας την εφαρμογή κατάλληλη για μελλοντικές βελτιώσεις και ενσωματώσεις χωρίς σημαντική αναδιάρθρωση του κώδικα. Πιθανές ωφέλιμες αλλαγές που μπορούν να γίνουν χωρίς να αλλάξει η βασική δομή της εφαρμογής έχουν ως εξής:

1. **Επέκταση σε περισσότερα μοντέλα τεχνητής νοημοσύνης:**

   Η χρήση του GPT-4o έγινε με REST API calls, γεγονός που καθιστά τη λύση ανεξάρτητη από συγκεκριμένο πάροχο ή μοντέλο. Στο μέλλον μπορούν να ενσωματωθούν άλλα μοντέλα όπως το Claude, Gemini και άλλα. Επίσης μπορούν να συνδεθούν και APIs για υπολογιστική όραση που βοηθούν στην προεπεξεργασία των εικόνων υπό εξέταση, όπως και custom μοντέλα που μπορούν να προκύψουν από fine-tuning. Εδώ θα ήταν χρήσιμο να εφαρμοστεί και μια διεπαφή επεξεργασίας των υπερπαραμέτρων όπως prompts, θερμοκρασίας κ.α. για περισσότερες δυνατότητες.

2. **Ενσωμάτωση με Pipelines CI/CD:**

   Η εφαρμογή μπορεί να ενταχθεί απευθείας σε διαδικασίες συνεχούς ολοκλήρωσης και ανάπτυξης (CI/CD). Έτσι, με κάθε νέα αλλαγή κώδικα μπορεί να γίνεται αυτόματη εκτέλεση των δοκιμών μέσω trigger, νυχτερινές εκτελέσεις αλλά και δημιουργία αναφορών σφαλμάτων και την καταχώρηση αυτών στο σύστημα ανάπτυξης του λογισμικού.

3. **Τροφοδότηση:**

   Η αξιολόγηση του μοντέλου γίνεται από τον χρήστη, όμως το σύστημα μπορεί να βελτιωθεί με προσθήκη μηχανισμού αξιολόγησης ανθρωπίνου χρήστη, όπου ο χρήστης επιβεβαιώνει εάν το μοντέλο έκανε σωστή πρόβλεψη. Έτσι μπορεί να χρησιμοποιηθούν αυτές οι ανατροφοδοτήσεις για fine-tuning του μοντέλου και βελτίωση των αποτελεσμάτων. Έτσι, μπορεί να έχουμε σταδιακή αυτοβελτίωση του εργαλείου χωρίς να διακόπτεται η χρήση του.

4. **Καλύτερη διαχείριση δοκιμών¨**

   Μπορεί εύκολα να προστεθούν δυνατότητες στην διαχείριση δοκιμών, όπως π.χ. κατηγοριοποίηση ειδών σφάλματος με ξεχωριστά prompts για καλύτερα αποτελέσματα, διαδικασία ελέγχου ακολουθίας στιγμιότυπων και όχι μόνο απομονωμένων οθονών, διαφορετικές εκδοχές συστημάτων κ.α.. Αυτό θα έκανε την εφαρμογή πιο εύχρηστη και δυναμική.

Αυτές οι δυνατότητες δείχνουν πως η παρούσα εφαρμογή, με την αρχιτεκτονική και τα εργαλεία που επιλέχθηκαν για την ανάπτυξή της, μπορεί από μια απλή εφαρμογή να μετατραπεί σε ένα δυναμικό σύστημα υποστήριξης δοκιμών βασισμένο σε AI,

προσαρμοσμένο σε διαφορετικές ανάγκες που μπορεί να υπάρξουν σε κάθε έργο. Η βάση λοιπόν που επιλέχθηκε και το πλαίσιο ανάπτυξης, ήταν κατάλληλες επιλογές.

# Κεφάλαιο 5



## Κεφάλαιο 5: Αξιολόγηση

Σε αυτό το κεφάλαιο, γίνεται η αξιολόγηση/εκτίμηση της αποτελεσματικότητας του εργαλείου αυτού. Το μοντέλο Chat-GPT4o χρησιμοποιείται «ωμά» χωρίς fine-tuning, κατά την απαίτηση των RSDs της εφαρμογής. Όμως, εντός των πλαισίων της, συμπεριλαμβάνονται διαφορετικά prompts που πιθανώς να βοηθήσουν στην καλύτερη εκτίμηση των περιπτώσεων και σε πιο ποιοτικά αποτελέσματα, ενώ παράλληλα γίνεται και προσαρμογή παραμέτρων που μπορεί να βοηθήσουν στην παραγωγή αποτελεσμάτων (π.χ. temperature). Ως προς την εκτίμηση της ποιότητας του εργαλείου, γίνονται χρήση τα μετρικά όπως περιεγράφηκαν στο 2.5 (Τρόπος Αξιολόγησης). Εφόσον αφορά σε ανίχνευση σφαλμάτων, έχουμε να κάνουμε με πραγματικά θετικά και αρνητικά, και λανθασμένα θετικά και αρνητικά. Επίσης, γίνεται χρήση της ακρίβειας (precision), ανάκλησης (recall) και μέτρησης F1 για την καλύτερη εξαγωγή αποτελεσμάτων. Εν τέλει, θα γίνει μια εκτίμηση της χρησιμότητας σε πραγματικές συνθήκες του εργαλείου, συγκρίνοντας το προβλεπόμενο επιχειρησιακό του κόστος σε σχέση με ανθρώπινο δυναμικό και το περιθώριο βελτίωσής του, αναφορικά με τις μετρήσεις που θα έχουν γίνει.

### 5.1 Μετρήσεις

Όπως αναφέρεται στο κεφάλαιο 2.5, τα δεδομένα χωρίστηκαν σε κάποιες βασικές κατηγορίες για καλύτερη εξαγωγή αποτελεσμάτων σε κάθε περίπτωση. Οι κατηγορίες είναι οι εξής: Χρωματισμός (Colouring), Διάταξη (Layout), Σχεδίαση (Design) και Γραμματοσειρές (Fonts), που είναι οι βασικές κατηγορίες που αφορούν σε συστήματα διαδικτύου. Επιπρόσθετα, έχει δημιουργηθεί και η κατηγορία mix, με δείγματα που

συμπεριλαμβάνουν σφάλματα από δύο ή περισσότερες κατηγορίες, αφού συνήθως σε ρεαλιστικές συνθήκες, μπορεί να υπάρχουν συνδυασμοί λαθών στην ίδια οθόνη. Ακόμα, υπάρχουν δέκα δείγματα χωρίς καμία διαφορά για να εκτιμηθεί επίσης το ενδεχόμενο το μοντέλο να φαντάζεται σφάλματα που δεν υπάρχουν, καθώς τέτοια μοντέλα είναι επιρρεπή σε σφάλματα «παραισθήσεων».**[19]**

Για δείγματα, έχουν δημιουργηθεί 100 συνολικά. Αυτό συνάδει με τις απαιτήσεις του RSD αλλά επίσης είναι αρκετά μεγάλος αριθμός να καλύψει τις ανάγκες εκτίμησης του εργαλείου. Πρώτον η καταμέτρηση και σύγκριση των αποτελεσμάτων του μοντέλου, καθώς παράγονται σε κείμενο, μπορεί να γίνει μόνο από άνθρωπο και άρα πρέπει να είναι αρκετά μικρό για να μπορούν να καταμετρηθούν σε πολλαπλές εκτελέσεις. Δεύτερον, καθώς αναφέρθηκε, θα γίνουν πολλαπλές εκτελέσεις με διάφορες παραμέτρους, άρα θα παραχθεί μια ολιστική εκτίμηση για τα δείγματα. Τρίτον, τα είδη οπτικών σφαλμάτων σε συστήματα διαδικτύου είναι συγκεκριμένα και αριθμήσιμα, άρα δεν χρειάζονται τόσο μεγάλα δεδομένα. Τέταρτον, δεν υπάρχουν περισσότερες από μερικές εκατοντάδες, στην χειρότερη περίπτωση, οθονών σε ένα σύστημα, άρα είναι και ένας αριθμός κοντά σε πραγματική χρήση. Εξάλλου, μερικά δείγματα περιλαμβάνουν πάνω από ένα σφάλμα. Εν τέλοις, δεν θα γίνει εκπαίδευση στο σύστημα, αλλά μόνο εκτίμηση, γι΄αυτό ο αριθμός είναι επαρκής (θα αντιστοιχούσε με χιλιάδα εκπαίδευσης). Τα δείγματα μαζί με την κατηγοριοποίησή τους εμφανίζονται στα παραρτήματα.

Ακολουθούν οι μετρήσεις σε κάθε περίπτωση που δοκιμάστηκε (όλες με τα ίδια δείγματα, αλλά με διαφορετικές παραμέτρους). Για σκοπούς χώρου και ευαναγνωσιμότητας, παραλείπεται ο πλήρης πίνακας του κάθε αποτελέσματος. Η έλλειψη αληθών αρνητικών επίσης, είναι διότι δεν θεώρησα αληθή αρνητικά για σφάλματα που δεν υπάρχουν. Αυτό γιατί, για όσα παραμένουν ίδια μεταξύ σχεδίου βάσης και screenshot, δεν μπορούν να κατηγοριοποιηθούν επακριβώς, και απλά θα ήταν μια γενική αρίθμηση. Επίσης δεν φέρουν επίδραση στην ακρίβεια, ανάκληση και μέτρηση F που είναι τα πιο σημαντικά σε αυτήν την περίπτωση.

**Συνθήκες εκτέλεσης:**

Η εντολή αναφέρεται στο κύριο prompt βάσει στο οποίο το μοντέλο αναγνωρίζει τι πρέπει να κάνει. Μία καλή εντολή μπορεί να βελτιώσει τα αποτελέσματα, ενώ εάν η

εντολή δεν είναι καλά καθορισμένη ενδέχεται να μειώσει την αποτελεσματικότητα των απαντήσεων.

Η θερμοκρασία [0.00, 1.00] σύμφωνα με το εγχειρίδιο της OpenAI, αναφέρεται στο πόσο «τυχαίες» ή δημιουργικές είναι οι απαντήσεις. Πιο μικρές τιμές παράγουν πιο ντετερμινιστικά και ακριβή αποτελέσματα. Οι πιο ψηλές έχουν ως αποτέλεσμα πιο δημιουργικές απαντήσεις.

Το top_p [0.00, 1.00], ή αλλιώς δειγματοληψία πυρήνα, είναι κατώφλι πιθανότητας. Είναι στρατηγική δειγματοληψίας που χρησιμοποιούν μοντέλα γλωσσών για την παραγωγή κειμένου. Καθορίζει πόσα από τα πιο πιθανά token το μοντέλο δικαιούται να «αναθεωρήσει» πριν την παραγωγή του επόμενου token για απάντηση **[20]**.

**Υποθέσεις:**

Καθώς η μεγαλύτερη θερμοκρασία αυξάνει την τυχαιότητα, είναι εύλογο να υποθέσουμε πως δεν θέλουμε ψηλή θερμοκρασία σαν παράμετρο, αφού θέλουμε αυστηρές και ντετερμινιστικές και καλώς ορισμένες προβλέψεις. Δεν θέλουμε δημιουργικές απαντήσεις, αλλά συνεπείς και εστιασμένες. Η συγκέντρωσή μας είναι σε αντικειμενικές διαφορές μεταξύ των εικόνων και συνεπώς θέλουμε ντετερμινιστική συμπεριφορά. Επίσης, θέλουμε όσο λιγότερες «παραισθήσεις» γίνεται. Ακόμα, αυτό συνάδει καλύτερα με το είδος δυαδικών απαντήσεων, αφού θέλουμε ένα «Σωστό» ή «Λάθος». Θέλουμε όμως και κάποια εφευρετικότητα για σφάλματα που είναι εκτός του συνήθους. Υπολογίζεται λοιπόν πως η θερμοκρασία γύρω στο 0.3 είναι η καλύτερη επιλογή.

Όσον αφορά το top_p, χρησιμοποιώντας ένα μεγαλύτερο top_p, κάνουμε την εξής «δήλωση» στο μοντέλο: «Συμπερίλαβε όλα τα πιθανά token εξόδου των οποίων η συνολική πιθανότητα αθροίζεται στο 95%, αγνοώντας τα λιγότερο πιθανά (~5%). Έτσι, το μοντέλο βάζει σε προτεραιότητα τις πιο πιθανές απαντήσεις. Ταυτόχρονα όμως θέλουμε μια προσαρμοστικότητα σε κάποιο βαθμό, γι' αυτό η καλύτερη επιλογή θα ήταν γύρω στο 0.95.

### *5.1.1 Περίπτωση 1*

**Παράμετροι:**

| | |
|---|---|
| Εντολή / Instruction | "Find differences. Ignore text contents, focus on design differences. If there are differences that are errors, in the first line write True, otherwise False. After that line, list the errors. I repeat, do not list text contents (such as different languages or different text details). Only state differences in fonts for the text." |
| Temperature/ Θερμοκρασία | temperature = 0.3 |
| Top_p | top_p = 0.95 |

**Αποτελέσματα:**

| Πραγματικά Επιτυχείς Δοκιμές | Πραγματικά Ανεπιτυχείς Δοκιμές | Επιτυχείς Δοκιμές | Ανεπιτυχείς Δοκιμές |
|---|---|---|---|
| 10 | 90 | 57 | 43 |

Ανά κατηγορία:

*Εδώ σε αντίθεση με τα επόμενα, δεν γίνεται καταμέτρηση ενός – ενός σφάλματος που υπάρχει σε κάθε δοκιμή, αλλά κατά πιο ποσοστό είχαμε σωστό αποτέλεσμα (δηλαδή passed όταν passed και failed όταν failed). Παρουσιάζονται σε ποσοστό [0,1].*

| Colouring | Layout | Design | Fonts | Mix | Same |
|---|---|---|---|---|---|
| **0.44** | **0.22** | **0.44** | **0.39** | **0.89** | **1.00** |

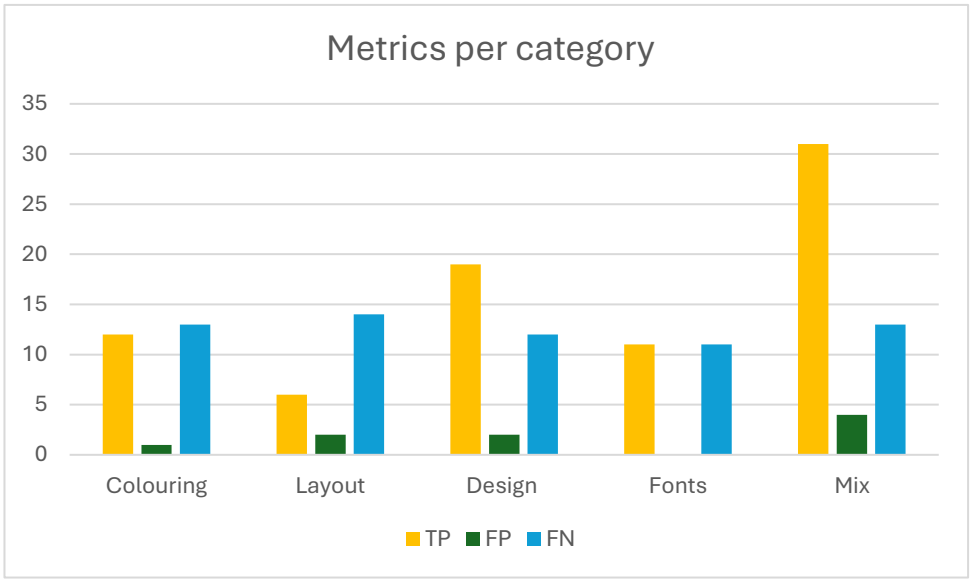
**Μετρήσεις**:

| | TP | TN | FP | FN | Recall | Precision | F |
|---|---|---|---|---|---|---|---|
| **Colouring** | **12.00** | **0.00** | **1.00** | **13.00** | **0.48** | **0.92** | **0.63** |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Layout** | **6.00** | **0.00** | **2.00** | **14.00** | **0.30** | **0.75** | **0.43** |
| **Design** | **19.00** | **0.00** | **2.00** | **12.00** | **0.61** | **0.90** | **0.73** |
| **Fonts** | **11.00** | **0.00** | **0.00** | **11.00** | **0.50** | **1.00** | **0.67** |
| **Mix** | **31.00** | **0.00** | **4.00** | **13.00** | **0.70** | **0.89** | **0.78** |
| **Total** | **79.00** | **10.00** | **9.00** | **63.00** | **0.56** | **0.90** | **0.69** |

**Heatmap**:

| | TP | TN | FP | FN | Recall | Precision | F |
|---|---|---|---|---|---|---|---|
| Colouring | 12.00 | 0.00 | 1.00 | 13.00 | 0.48 | 0.92 | 0.63 |
| Layout | 6.00 | 0.00 | 2.00 | 14.00 | 0.30 | 0.75 | 0.43 |
| Design | 19.00 | 0.00 | 2.00 | 12.00 | 0.61 | 0.90 | 0.73 |
| Fonts | 11.00 | 0.00 | 0.00 | 11.00 | 0.50 | 1.00 | 0.67 |
| Mix | 31.00 | 0.00 | 4.00 | 13.00 | 0.70 | 0.89 | 0.78 |
| Total | 79.00 | 10.00 | 9.00 | 63.00 | 0.56 | 0.90 | 0.69 |



**Χρόνος εκτέλεσης**: 60173ms

### *5.1.2 Περίπτωση 2*

**Παράμετροι:**

| | |
|---|---|
| Εντολή / Instruction | "Find differences. Do not focus much on text contents, focus on design differences. If there are differences that are errors, in the first line write True, otherwise False. After that line, list the errors. Only state differences in fonts for the text. Try examining small differences in color hues, notice the layout in text contents and ordering, and also small design differences." |
| Temperature/ Θερμοκρασία | temperature = 0.3 |
| Top_p | top_p = 0.95 |

**Αποτελέσματα:**

| Πραγματικά Επιτυχείς Δοκιμές | Πραγματικά Ανεπιτυχείς Δοκιμές | Επιτυχείς Δοκιμές | Ανεπιτυχείς Δοκιμές |
|---|---|---|---|
| 10 | 90 | 51 | 49 |

Ανά κατηγορία:

| Colouring | Layout | Design | Fonts | Mix | Same |
|---|---|---|---|---|---|
| 0.50 | 0.28 | 0.61 | 0.39 | 0.94 | 1.00 |

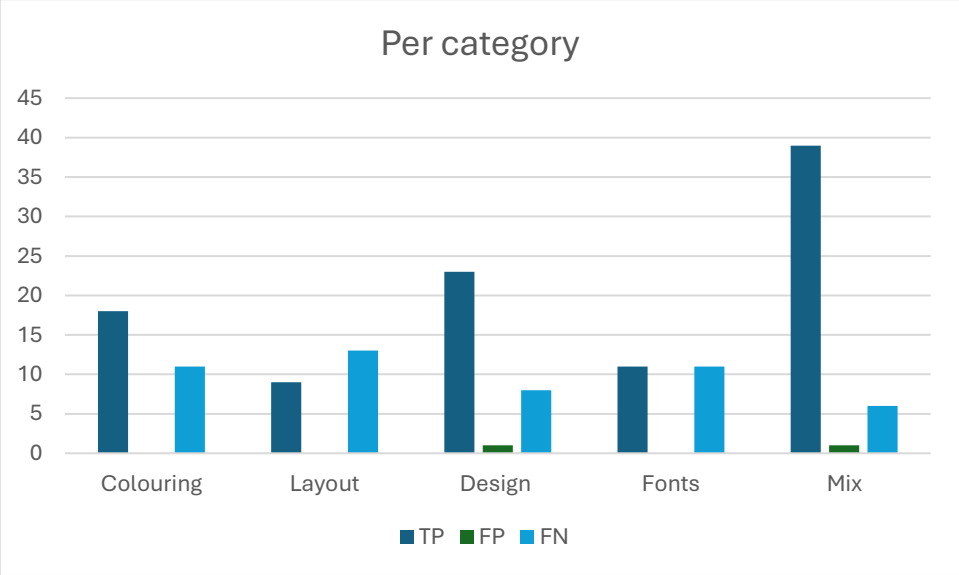
**Μετρήσεις:**

| | TP | TN | FP | FN | Recall | Precision | F |
|---|---|---|---|---|---|---|---|

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Colouring | 18.00 | 0.00 | 0.00 | 11.00 | 0.62 | 1.00 | 0.77 |
| Layout | 9.00 | 0.00 | 0.00 | 13.00 | 0.41 | 1.00 | 0.58 |
| Design | 23.00 | 0.00 | 1.00 | 8.00 | 0.74 | 0.96 | 0.84 |
| Fonts | 11.00 | 0.00 | 0.00 | 11.00 | 0.50 | 1.00 | 0.67 |
| Mix | 39.00 | 0.00 | 1.00 | 6.00 | 0.87 | 0.98 | 0.92 |
| Total | 100.00 | 10.00 | 2.00 | 49.00 | 0.67 | 0.98 | 0.80 |

Heatmap:

| | TP | TN | FP | FN | Recall | Precision | F |
|---|---|---|---|---|---|---|---|
| Colouring | 18.00 | 0.00 | 0.00 | 11.00 | 0.62 | 1.00 | 0.77 |
| Layout | 9.00 | 0.00 | 0.00 | 13.00 | 0.41 | 1.00 | 0.58 |
| Design | 23.00 | 0.00 | 1.00 | 8.00 | 0.74 | 0.96 | 0.84 |
| Fonts | 11.00 | 0.00 | 0.00 | 11.00 | 0.50 | 1.00 | 0.67 |
| Mix | 39.00 | 0.00 | 1.00 | 6.00 | 0.87 | 0.98 | 0.92 |
| Total | 100.00 | 10.00 | 2.00 | 49.00 | 0.67 | 0.98 | 0.80 |




**Χρόνος εκτέλεσης**: 59664ms

### *5.1.3 Περίπτωση 3*

**Παράμετροι:**

| | |
|---|---|
| Εντολή / Instruction | "Find differences. Do not focus much on text contents, focus on design differences. If there are differences that are errors, in the first line write True, otherwise False. After that line, list the errors. Only state differences in fonts for the text. Try examining small differences in color hues, notice the layout in text contents and ordering, and also small design differences." |
| Temperature/ Θερμοκρασία | temperature = 0.1 |
| Top_p | top_p = 1 |

**Αποτελέσματα:**

| Πραγματικά Επιτυχείς Δοκιμές | Πραγματικά Ανεπιτυχείς Δοκιμές | Επιτυχείς Δοκιμές | Ανεπιτυχείς Δοκιμές |
|---|---|---|---|
| 10 | 90 | 52 | 48 |

Ανά κατηγορία:

| Colouring | Layout | Design | Fonts | Mix | Same |
|---|---|---|---|---|---|
| 0.56 | 0.28 | 0.61 | 0.33 | 0.89 | 1.00 |

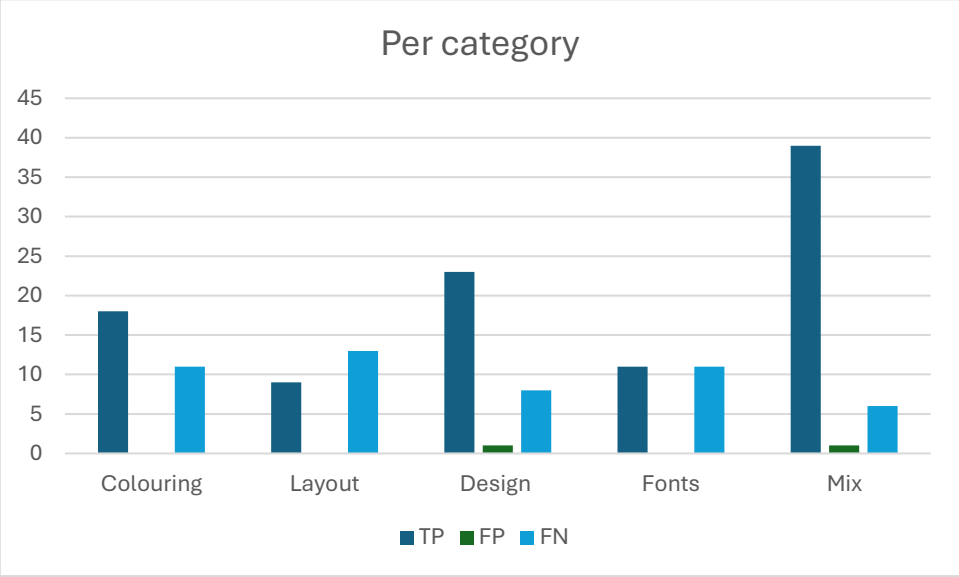
**Μετρήσεις:**

| | TP | TN | FP | FN | Recall | Precision | F |
|---|---|---|---|---|---|---|---|
| Colouring | 20.00 | 0.00 | 0.00 | 7.00 | 0.74 | 1.00 | 0.85 |
| Layout | 8.00 | 0.00 | 0.00 | 14.00 | 0.36 | 1.00 | 0.53 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Design | 23.00 | 0.00 | 9.00 | 8.00 | 0.74 | 0.72 | 0.73 |
| Fonts | 8.00 | 0.00 | 2.00 | 15.00 | 0.35 | 0.80 | 0.48 |
| Mix | 36.00 | 1.00 | 5.00 | 6.00 | 0.86 | 0.88 | 0.87 |
| Total | 95.00 | 10.00 | 16.00 | 50.00 | 0.66 | 0.86 | 0.74 |

Heatmap:

| | TP | TN | FP | FN | Recall | Precision | F |
|---|---|---|---|---|---|---|---|
| Colouring | 20.00 | 0.00 | 0.00 | 7.00 | 0.74 | 1.00 | 0.85 |
| Layout | 8.00 | 0.00 | 0.00 | 14.00 | 0.36 | 1.00 | 0.53 |
| Design | 23.00 | 0.00 | 9.00 | 8.00 | 0.74 | 0.72 | 0.73 |
| Fonts | 8.00 | 0.00 | 2.00 | 15.00 | 0.35 | 0.80 | 0.48 |
| Mix | 36.00 | 1.00 | 5.00 | 6.00 | 0.86 | 0.88 | 0.87 |
| Total | 95.00 | 10.00 | 16.00 | 50.00 | 0.66 | 0.86 | 0.74 |




**Χρόνος εκτέλεσης**: 59653ms

### *5.1.4 Περίπτωση 4*

**Παράμετροι:**

| | |
|---|---|
| Εντολή / Instruction | "Find differences. Do not focus much on text contents, focus on design differences. If there are differences that are errors, in the first line write True, otherwise False. After that line, list the errors. Only state differences in fonts for the text. Try examining small differences in color hues, notice the layout in text contents and ordering, and also small design differences." |
| Temperature/ Θερμοκρασία | temperature = 0.8 |
| Top_p | top_p = 0.95 |

**Αποτελέσματα:**

| Πραγματικά Επιτυχείς Δοκιμές | Πραγματικά Ανεπιτυχείς Δοκιμές | Επιτυχείς Δοκιμές | Ανεπιτυχείς Δοκιμές |
|---|---|---|---|
| 10 | 90 | 61 | 39 |

Ανά κατηγορία:

| Colouring | Layout | Design | Fonts | Mix | Same |
|---|---|---|---|---|---|
| 0.33 | 0.28 | 0.56 | 0.28 | 0.72 | 1.00 |

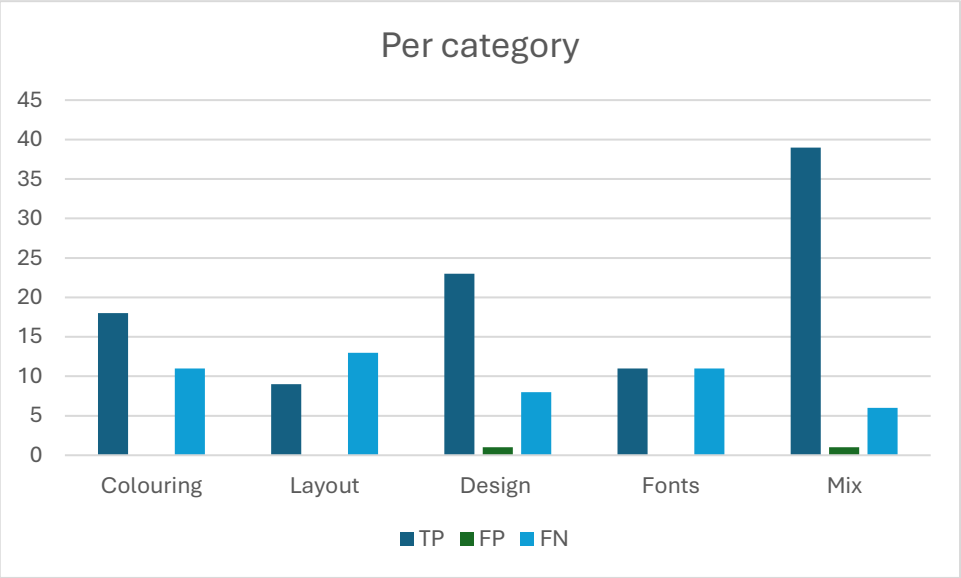
**Μετρήσεις:**

| | TP | TN | FP | FN | Recall | Precision | F |
|---|---|---|---|---|---|---|---|
| Colouring | 12.00 | 0.00 | 1.00 | 15.00 | 0.44 | 0.92 | 0.60 |
| Layout | 9.00 | 0.00 | 2.00 | 14.00 | 0.39 | 0.82 | 0.53 |
| Design | 19.00 | 0.00 | 1.00 | 12.00 | 0.61 | 0.95 | 0.75 |
| Fonts | 9.00 | 0.00 | 6.00 | 14.00 | 0.39 | 0.60 | 0.47 |
| Mix | 27.00 | 1.00 | 4.00 | 15.00 | 0.64 | 0.87 | 0.74 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Total | 76.00 | 10.00 | 14.00 | 70.00 | 0.52 | 0.84 | 0.64 |

Heatmap:

| | TP | TN | FP | FN | Recall | Precision | F |
|---|---|---|---|---|---|---|---|
| Colouring | 12.00 | 0.00 | 1.00 | 15.00 | 0.44 | 0.92 | 0.60 |
| Layout | 9.00 | 0.00 | 2.00 | 14.00 | 0.39 | 0.82 | 0.53 |
| Design | 19.00 | 0.00 | 1.00 | 12.00 | 0.61 | 0.95 | 0.75 |
| Fonts | 9.00 | 0.00 | 6.00 | 14.00 | 0.39 | 0.60 | 0.47 |
| Mix | 27.00 | 1.00 | 4.00 | 15.00 | 0.64 | 0.87 | 0.74 |
| Total | 76.00 | 10.00 | 14.00 | 70.00 | 0.52 | 0.84 | 0.64 |




**Χρόνος εκτέλεσης**: 60432ms

### *5.1.5 Περίπτωση 5*

**Παραμέτροι:**

| | |
|---|---|
| Εντολή / Instruction | "Find differences. Do not focus much on text contents, focus on design differences. If there are differences that are errors, in the first line write True, otherwise False. After that line, list the errors. Only state differences in fonts for the text. Try examining small differences in color hues, notice the layout in text contents and ordering, and also small design differences." |
| Temperature/ Θερμοκρασία | temperature = 0.3 |
| Top_p | top_p = 0.3 |

**Αποτελέσματα:**

| Πραγματικά Επιτυχείς Δοκιμές | Πραγματικά Ανεπιτυχείς Δοκιμές | Επιτυχείς Δοκιμές | Ανεπιτυχείς Δοκιμές |
|---|---|---|---|
| 10 | 90 | 59 | 41 |

Ανά κατηγορία:

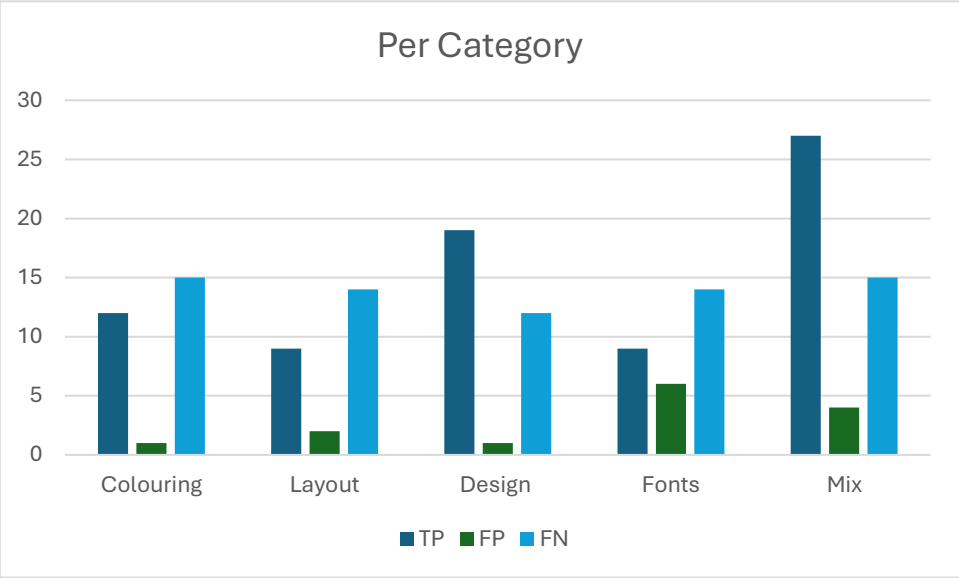
| Colouring | Layout | Design | Fonts | Mix | Same |
|---|---|---|---|---|---|
| 0.44 | 0.28 | 0.56 | 0.28 | 0.72 | 1.00 |

**Μετρήσεις:**

| | TP | TN | FP | FN | Recall | Precision | F |
|---|---|---|---|---|---|---|---|
| Colouring | 15.00 | 0.00 | 1.00 | 11.00 | 0.58 | 0.94 | 0.71 |
| Layout | 7.00 | 0.00 | 4.00 | 16.00 | 0.30 | 0.64 | 0.41 |
| Design | 22.00 | 0.00 | 3.00 | 9.00 | 0.71 | 0.88 | 0.79 |
| Fonts | 10.00 | 0.00 | 5.00 | 13.00 | 0.43 | 0.67 | 0.53 |
| Mix | 25.00 | 1.00 | 4.00 | 17.00 | 0.60 | 0.86 | 0.70 |
| Total | 79.00 | 10.00 | 17.00 | 66.00 | 0.54 | 0.82 | 0.66 |

Heatmap:

| | TP | TN | FP | FN | Recall | Precision | F |
|---|---|---|---|---|---|---|---|
| Colouring | 15.00 | 0.00 | 1.00 | 11.00 | 0.58 | 0.94 | 0.71 |
| Layout | 7.00 | 0.00 | 4.00 | 16.00 | 0.30 | 0.64 | 0.41 |
| Design | 22.00 | 0.00 | 3.00 | 9.00 | 0.71 | 0.88 | 0.79 |
| Fonts | 10.00 | 0.00 | 5.00 | 13.00 | 0.43 | 0.67 | 0.53 |
| Mix | 25.00 | 1.00 | 4.00 | 17.00 | 0.60 | 0.86 | 0.70 |
| Total | 79.00 | 10.00 | 17.00 | 66.00 | 0.54 | 0.82 | 0.66 |




**Χρόνος εκτέλεσης**: 61223ms

## 5.2 Συγκρίσεις και Αποτελέσματα

Εν πρώτοις, από μία πρόχειρη ματιά τα αποτελέσματα δεν φαίνονται ιδιαίτερα ποιοτικά, αφού έχουμε πολλές περιπτώσεις λανθασμένων αρνητικών. Στην καλύτερη περίπτωση αυτής της μετρικής, στην 2, είχαμε 51 «passed» και 49 «failed», έναντι των 10 και 90 που θα έπρεπε να υπάρχουν. Όμως, αυτοί οι αριθμοί δεν αντικατοπτρίζουν καλά την πραγματική εικόνα των αποτελεσμάτων. Αρχικά, εδώ φαίνονται συνολικά οι αριθμοί και δεν είναι χωρισμένοι ανά κατηγορία. Ας πούμε, η κατηγορία διάταξης των στοιχείων, σε όλες τις περιπτώσεις έχει κάτω από 30% ανιχνευσιμότητα, όμως άλλες κατηγορίες όπως π.χ. αυτό των σχεδίων έχουν πάνω από 50%. Ταυτόχρονα, οι «δύσκολες» περιπτώσεις,

δηλαδή αυτές που περιέχουν μικρές και λίγες διαφορές, χαλούν τα αποτελέσματα. Ας πούμε, το colouring_1, περιέχει μόνο ένα σφάλμα χρωματισμού το οποίο είναι μια ελαφρώς διαφορετική χροιά του πορτοκαλιού, ενώ το layout_10 απλά έχει μια ελαφρώς διαφορετική τοποθέτηση του κουμπιού αναζήτησης. Ενώ, την ίδια ώρα, άλλες περιπτώσεις όπως το colouring_18, περιέχουν πιο πολλά σφάλματα τα οποία ανιχνεύονται με ακρίβεια. Ας δούμε λοιπόν τις υπόλοιπες μετρικές, εξάγοντας όσα συμπεράσματα είναι δυνατόν.

Σημαντικό εδώ, είναι να δούμε κατά πόσον έχουν επιβεβαιωθεί οι υποθέσεις μας, όπως αναφέρθηκαν προηγουμένως. Όσον αφορά στην θερμοκρασία, πράγματι οι περιπτώσεις 1 και 2, όπου η θερμοκρασία ήταν πιο χαμηλή, αμφότερες είχαν καλύτερα από τα αποτελέσματα της περίπτωσης 4 που είχε υψηλή θερμοκρασία. Στο 1 και στο 2 οι συνολικές ανάκληση, ακρίβεια και F-score ήταν 0.67, 0.98 και 0.79 και 0.55, 0.89, 0.68 αντίστοιχα, ενώ στην $4^{η}$ περίπτωση ήταν 0.52, 0.84 και 0.64 που φαίνεται ότι είχαμε σημαντική πτώση. Επίσης το μέγιστο F-score στην περίπτωση υψηλής θερμοκρασίας ήταν το 0.739 στην περίπτωση των μεικτών σφαλμάτων, ενώ στην $1^{η}$ είχαμε μέγιστο 0.78 στην ίδια κατηγορία και στην $2^{η}$ είχαμε 0.91 (σημαντική πτώση!). Ταυτόχρονα, το πιο ψηλό ποσοστό ανιχνευσιμότητας στην $4^{η}$ περίπτωση ήταν το 72.2% ενώ στις άλλες κατηγορίες 88.8% και 94.4% αντίστοιχα.

Όσον αφορά το top_p, είχαμε υποθέσει πως ένα πιο ψηλό top_p είναι καλύτερο, εδώ χρειάζεται σύγκριση του 1,2 και 5. Στην $5^{η}$ περίπτωση, επίσης το μέγιστο ποσοστό ανιχνευσιμότητας ήταν 72.2%, ενώ συνολικά είχαμε 0.54 ανάκληση, 0.82 ακρίβεια και 0.65 F-score, βάζοντάς το επίσης σε χειρότερη θέση από την κατηγορία.

Στην $3^{η}$ κατηγορία γίνεται κάτι διαφορετικό. Εδώ βάζουμε πιο ακραίες τιμές στην θερμοκρασία. Η θερμοκρασία πέφτει κατά πολύ (0.1 έναντι του 0.3) ενώ το top_p μπαίνει στο μέγιστο του. Το prompt που χρησιμοποιείται είναι το ίδιο με αυτό της κατηγορίας 2. Εδώ βλέπουμε πως συνολικά έχουμε ανάκληση 0.66, ακρίβεια 0.86 και F-score 0.74. Αυτά είναι αρκετά μικρότερα (ιδίως η συνολική ακρίβεια) από την $2^{η}$ περίπτωση. Αυτό επιβεβαιώνει περαιτέρω τις υποθέσεις μας, καθώς δεν θέλουμε τις ακρότατες τιμές, αλλά θέλουμε να παραμείνει μια ισορροπία. Δεν θέλουμε να είναι εντελώς αυστηρό το μοντέλο, και θέλουμε να υπάρχει μια ευελιξία στο πόσο «περίεργο» ή «ασυνήθιστο» μπορεί να είναι ένα σφάλμα.

Όσον αφορά τα prompts, βλέπουμε, με βάση τις πληροφορίες που αναφέρθηκαν πιο πάνω, πως η δεύτερη περίπτωση ήταν πιο αποτελεσματική μετά την αλλαγή των prompt. Ας εξετάσουμε τις διαφορές τους. Το πρώτο ήταν «Βρες διαφορές. Αγνόησε τα περιεχόμενα του κειμένου, εστίασε στις διαφορές σχεδιασμού. Εάν υπάρχουν διαφορές που είναι σφάλματα, στην πρώτη γραμμή γράψε Αληθές, αλλιώς Λανθασμένο. Μετά από αυτήν την γραμμή, καταμέτρησε τα σφάλματα. Επαναλαμβάνω, μην αναφέρεις τα περιεχόμενα του κειμένου, (όπως διαφορετικές γλώσσες ή λεπτομέρειες κειμένου). Επικεντρώσου μόνο σε διαφορές γραμματοσειράς για το κείμενο». Με αυτό το prompt παρατηρήθηκαν κακά αποτελέσματα για τις κατηγορίες χρωματισμού, διάταξης και γραμματοσειρών. Αυτό οδήγησε στην αναθεώρηση του prompt. Θεωρήθηκε καλύτερο να μην γίνεται αναφορά μόνο στα σχεδιαστικά λάθη, αλλά και στην σειρά των κειμένων ή λεπτότερων διαφορών. Έτσι προστέθηκε το εξής στο prompt: «Προσπάθησε να εξετάσεις μικρές διαφορές στις χροιές χρωμάτων, να προσέξεις την διάταξη στα περιεχόμενα κειμένου και στην σειρά τους.» Πράγματι, αυξήθηκε η ανιχνευσιμότητα σε όλες τις κατηγορίες και αυξήθηκε η ανάκληση. Για παράδειγμα, στην $1^{η}$ περίπτωση η πιο μικρή ανάκληση στην διάταξη ήταν 0.3 ενώ στην $2^{η}$ ήταν 0.4. Σημαντικό είναι επίσης να αναφερθεί πως όλες οι μετρήσεις της $2^{ης}$ κατηγορίας, η οποία ήταν και γενικώς η καλύτερη, ήταν καλύτερες από αυτές της $1^{ης}$ κατηγορίας. Βλέπουμε συνεπώς πως πιο σαφείς οδηγίες οδηγούν και σε καλύτερα αποτελέσματα.

Κάτι το ενδιαφέρον, είναι πως ενώ η ανάκληση ήταν σχετικά χαμηλή πολλές φορές, η ακρίβεια έμεινε ψηλά. Η χειρότερη ακρίβεια που καταμετρήθηκε ήταν 0.6 στην $4^{η}$ περίπτωση. Στην δεύτερη, συνολική ακρίβεια ήταν 0.98 με 3 κατηγορίες να έχουν καθαρό 1. Ταυτόχρονα, σε καμία από τις εκτελέσεις δεν είχαμε λανθασμένο αποτέλεσμα στα 10 δείγματα που δεν περιέχουν λάθος. Αυτό σημαίνει πως το μοντέλο έχει χαμηλά τα φαινόμενα της «φαντασίας», και εάν αναφέρει κάποιο λάθος τότε στις πλείστες των περιπτώσεων πράγματι θα είναι λάθος. Η χαμηλή ακρίβεια που παρατηρήθηκε επίσης ήταν στην περίπτωση με ψηλή θερμοκρασία, και άρα δημιουργικότητα.

Όμως, η ανάκληση, ακόμα και στην $2^{η}$ περίπτωση, είχε χαμηλές μετρήσεις. Στην διάταξη ήταν 0.4 και στις γραμματοσειρές 0.5. Γενικώς, και στις τρεις περιπτώσεις, λανθασμένα αρνητικά είχαμε κυρίως στον χρωματισμό, διάταξη και γραμματοσειρές, με τα χειρότερα να βρίσκονται στην διάταξη. Αυτό αντικατοπτρίζεται και με το F-score. Η χαμηλή ανάκληση, δηλαδή η αγνόηση αρκετών σφαλμάτων, ίσως να οφείλεται στο ότι τα

generative μοντέλα, και ιδιαίτερα το ChatGPT είναι εκπαιδευμένα να αποφεύγουν σφάλματα σε ακραίες περιπτώσεις. Αυτό συνάδει και με τον χρωματισμό, όπου σε μικρές διαφορές χρωματισμού, δεν ανέφερε το σφάλμα. Επίσης, όταν υπάρχουν πολλαπλά λάθη, έχουμε καλύτερα αποτελέσματα, πράγμα που σημαίνει πως μάλλον αυτό είναι μια περίπτωση συγκεντρωτικού αποτελέσματος, έτσι ακόμα και μικρά σφάλματα, εάν τοποθετούνται μαζί με άλλα, θα παρατηρούνται από το μοντέλο. Επίσης, στις κατηγορίες γραμματοσειρών, χρωματισμού και διάταξης, υπήρχαν οι πιο «μικρές» διαφορές, όπου μια μικρή διαφορά μεταξύ χρώματος, ή μια κάπως διαφορετική γραμματοσειρά, ή λίγη διαφορά στα διαστήματα, ήταν δύσκολο να ανιχνευτούν από το μοντέλο. Αυτό σημαίνει πως έχει μια δυσκολία στα λίγα και μικρά λάθη.

Όπως είχε υποτεθεί, πράγματι η $2^{η}$ περίπτωση είναι η καλύτερη στις μετρήσεις. Σε μία μόνο περίπτωση όμως, έχει χειρότερα αποτελέσματα. Στην περίπτωση του χρωματισμού, στην $5^{η}$ περίπτωση βλέπουμε 0.57 ανάκληση και 0.93 ακρίβεια, ενώ στην $3^{η}$ 0.74 ανάκληση και 1 ακρίβεια. Η $3^{η}$ περίπτωση είναι αυτή με την πιο χαμηλή θερμοκρασία και πιο μεγάλο top_p. Αυτό ίσως να αυξάνει την αυστηρότητα του μοντέλου στην ανίχνευση μικρών διαφορών στο χρώμα, έχοντας όμως αντίκτυπο στην αύξηση άλλων λανθασμένων αποτελεσμάτων (π.χ. τα λανθασμένα θετικά είναι οκτώ φορές περισσότερα συνολικά).

# Κεφάλαιο 6



## Κεφάλαιο 6: Επιχειρησιακή Σύνοψη και Συμπεράσματα

### 6.1 Επιχειρησιακή Σύνοψη

Στην παρούσα της μορφή, η εφαρμογή δεν μπορεί να χρησιμοποιηθεί σε επιχειρησιακό επίπεδο. Παρότι έχει υψηλή ακρίβεια, κάτι το οποίο είναι πολύ σημαντικό σε πραγματικές συνθήκες, η χαμηλή ανάκληση στις περιπτώσεις όπου υπάρχουν λίγα ή ένα σφάλμα σε κάθε οθόνη, και αυτά δεν είναι μεγάλα, δεν μπορεί να χρησιμοποιηθεί πιστότητα. Από αυτήν την άποψη, ίσως μπορεί να χρησιμοποιηθεί σαν εργαλείο επιτάχυνσης εύρεσης σοβαρών οπτικών σφαλμάτων, καθώς όπως εξηγήθηκε στο 4.2, εάν τα σφάλματα είναι σημαντικά σε αριθμό και ποιότητα, θα αναφερθούν με υψηλή ακρίβεια.

Όσον αφορά το κόστος, σύμφωνα με το OpenAI **[21]** οι υπολογισμοί θα έχουν ως εξής:

-Είσοδος: 2 εικόνες 1920x1080. Περίπου 1445 tokens ανά εικόνα. Άρα 2x1445+500 περίπου για το κείμενο. Συνολικά 3390 * $5/1M input tokens, περίπου 2 Σεντ του δολαρίου ανά είσοδο.

-Έξοδος: 800 tokens. Άρα συνολικά 0.02095 Σεντ του δολαρίου ανά σενάριο δόκιμης.

Αυτό σημαίνει πως σε ένα Project 2-3 χρόνων με περίπου 500 οθόνες και περίπου 100 εκτελέσεις επαναληπτικότητας, θα υπήρχε περίπου κόστος των 1000 δολαρίων. Εάν υποτεθεί πως ένας άνθρωπος θέλει περίπου 5-10 λεπτά για κάθε οθόνη, με μισθό περίπου 20 ευρώ την ώρα, θα υπήρχε κόστος περίπου 83k για την ίδια εκτέλεση. Άρα, από οικονομική άποψη, αυτό το εργαλείο έχει προοπτική εάν υπάρξει σημαντική βελτίωση στην ανάκληση.

Όσον αφορά στην εκτέλεση, είχαμε μέσο όρο 60229ms για 100 δοκιμές. Αυτό σημαίνει ότι κάθε δοκιμή απαιτεί περίπου 0.6s που πληροί τις απαιτήσεις των RSDs.

Συνεπώς, στην παρούσα του φάση, θα μπορούσε στην καλύτερη των περιπτώσεων να χρησιμοποιηθεί σαν εργαλείο γρήγορης εκτέλεσης ούτως ώστε να καθαρίζουν γρήγορα τα μεγάλα σφάλματα, χωρίς περισσότερη χρήση.

## 6.2 Συμπεράσματα

Η μελέτη ανέδειξε σημαντικά ευρήματα σχετικά με τις επιδόσεις του μοντέλου για οπτικό έλεγχο. Υπήρξαν ιδιαίτερα ψηλές τιμές ακρίβειας, που σημαίνει ότι συνήθως εντοπίζεται κάποιο σφάλμα, αυτό πράγματι υπάρχει. Όμως, οι χαμηλές τιμές ανάκλησής του το καθιστούν ως μη – ικανό εργαλείο για αυτοματοποίηση αυτής της δουλείας στην παρούσα φάση χωρίς να αποκλείουν εντελώς το μελλοντικό ενδεχόμενο χρήσης του. Το φαινόμενο αυτό εντοπίστηκε κυρίως στις κατηγορίες χρωματισμού, διάταξης και γραμματοσειρών όπου οι διαφορές ήταν μικρές και δύσκολα ανιχνεύσιμες. Φάνηκε, την ίδια ώρα, πως υπάρχει μια ευαισθησία προς το συγκεντρωτικό φαινόμενο σφαλμάτων, τα οποία τυγχάνουν καλύτερης ανιχνευσιμότητας.

Ταυτόχρονα, αναδείχθηκε η επίδραση των υπερπαραμέτρων (θερμοκρασίας και top_p) και πως αυτές πρέπει να ληφθούν σοβαρά υπόψη στην ανάπτυξη τέτοιων εφαρμογών. Η θερμοκρασία πρέπει να είναι χαμηλή, αλλά όχι εντελώς, και το top_p υψηλό αλλά όχι στο μέγιστό του. Αυτό διατηρεί μια αυστηρότητα και συνέπεια που απαιτείται στην ανίχνευση σφαλμάτων. Επίσης, φάνηκε πως η αναδιατύπωση των οδηγιών είχε καθοριστικό ρόλο στη βελτίωση της απόδοσης του μοντέλου.

Εν τέλει, σε θεωρητικό επίπεδο, θα μπορούσε μελλοντικά η εφαρμογή αυτή να έχει πραγματική χρήση στην βιομηχανία. Οικονομικά, το κόστος χρήσης είναι σημαντικά χαμηλότερο από το ανθρώπινο αντίστοιχο – και ακόμα και να μείωνε σε 10-20% το απαιτούμενο ανθρώπινο δυναμικό θα βοηθούσε. Επίσης, η ταχύτητα εκτέλεσης είναι ικανά μικρή για να μπορεί να ενταχθεί σε CI/CD Pipeline (συνεχούς ανάπτυξης). Όμως, στην πράξη, αυτό δεν μπορεί να συμβεί, λόγω της ιδιαίτερα χαμηλής ανάκλησης σε διάφορες κατηγορίες.

Εν ολίγοις, η μελέτη ανέδειξε ένα εργαλείο με δυναμικό και καλή ακρίβεια, χωρίς παραισθήσεις, αλλά με περιορισμούς λόγω της χαμηλής ανάκλησης, κάνοντας την εφαρμογή προς το παρόν μη αξιόπιστη.

## 6.3 Μελλοντική Εργασία

Παρότι η παρούσα προσέγγιση έχει ενδιαφέροντα αποτελέσματα στο να ανιχνεύει σημαντικά οπτικά σφάλματα με υψηλή ακρίβεια, υπάρχουν αρκετές κατευθύνσεις μελλοντικής εργασίας που μπορούν να βελτιώσουν την απόδοση του συστήματος, για να διευθετηθούν τα προβλήματα με την χαμηλή ανάκληση.

Πρώτη και κύρια μέθοδος είναι το fine-tuning. Με αυτό μπορεί να γίνει βελτίωση του μοντέλου χρησιμοποιώντας δεδομένα συγκεκριμένα σε αυτόν τον τομέα. Το παρόν μοντέλο είναι γενικής χρήσεως και δεν έχει αρκετή ευαισθησία στις λεπτές διαφορές που προκύπτουν στους ελέγχους γραφικών διεπαφών, όπως ελαφρές αλλαγές στην διάταξη, γραμματοσειρά ή χρωματισμό. Εάν γίνει ένα καλό fine-tuning μπορεί το μοντέλο να αποκτήσει καλύτερη κατανόηση αυτών των κατηγοριών.

Ακόμα, μπορεί να γίνει πειραματισμός με την κατηγοριοποίηση των δοκιμών ελέγχου με βάση τις κατηγορίες που έχουν αναφερθεί ή άλλες, για να χρησιμοποιούν διαφορετικές υπερπαραμέτρους ή οδηγίες. Για παράδειγμα, ένα σενάριο δοκιμής επικεντρωμένο στην διάταξη θα είχε άλλη στρατηγική εκτίμησης από αυτό που είναι τοποθετημένο στον χρωματισμό. Με την εισαγωγή μιας ελαφριάς φάσης κατηγοριοποίησης – είτε χειρωνακτικής ή υποβοηθούμενη μέσω μηχανικής μάθησης – θα μπορούσαν να βελτιωθούν τα αποτελέσματα, αν και θα αυξανόταν το επιχειρησιακό κόστος.

Επιπρόσθετα, θα μπορούσε να γίνει εξερεύνηση εναλλακτικών μοντέλων. Η παρούσα διασύνδεση με το GPT θα μπορούσε να αντικατασταθεί με άλλα μοντέλα, όπως το Claude, Mistral κ.α. Μια συγκριτική ανάλυση μεταξύ τους θα μπορούσε να εντοπίσει μοντέλα που έχουν καλύτερες αποδόσεις σε διαφορετικές κατηγορίες σφαλμάτων, ή που ανταποκρίνονται καλύτερα σε αλλαγές οδηγιών. Μπορεί επίσης να μειώσει το επιχειρησιακό κόστος.

Σε συνδυασμό, αυτές οι κατευθύνσεις, το fine – tuning, οι προσαρμοσμένες οδηγίες μέσω κατηγοριοποίησης των σεναρίων δοκιμών και σύγκριση μεταξύ διαφόρων μοντέλων, θα μπορούσαν να βελτιώσουν σημαντικά τις αποδόσεις της εφαρμογής. Ίσως σύντομα, προϊόντα αυτόματης ανίχνευσης οπτικών σφαλμάτων, με χρήση LLMs, ή ακόμα άλλων

αρχιτεκτονικών που περιλαμβάνουν και άλλες τεχνικές επεξεργασίας εικόνων, να είναι μια πραγματικότητα που κάνουν την δοκιμή και την ανάπτυξη λογισμικού πολύ γρηγορότερη και ευχάριστη.

# Βιβλιογραφία


**[1] M. Junaid και S. Z. Ahmed, «A Reseach study on importance of Testing and Quality Assurance in Software Development life cycle (SDLC) Models, Volume 12, Issue 2,» *International Journal of Scientific & Engineering Research,* 2021.**

**[2] D. Talby, A. Keren, O. Hazzan και Y. Dubinsky, «Agile software testing in a large-scale project,» *IEEE Software,* τόμ. 23, αρ. 4, pp. 30-37, 2006.**

**[3] D. Mohammad, K. M, P. K. και M. M.V., «Benefits and limitations of automated software testing: Systematic literature review and practitioner survey,» *2012 7th International Workshop on Automation of Software Test (AST),* 2012.**

**[4] L. Z., L. C., C. C., W. J., C. M., W. B., W. Y., H. J. και W. Q., «Seeing is Believing: Vision-driven Non-crash Functional Bug Detection for Mobile Apps,» *https://arxiv.org,* 2024. Available: https://arxiv.org/abs/2407.03037v2**

**[5] OpenAI, «OpenAI,» OpenAI, May 2024. [Ηλεκτρονικό]. Available: https://openai.com/index/hello-gpt-4o/.**

**[6] Markets and Markets , «Automation Testing Market by Offering (Testing Types (Static Testing and Dynamic Testing) and Services) Endpoint Interface (Mobile, Web, Desktop, and Embedded Software), Vertical (BFSI, Automotive, IT & ITeS) and Region - Global Forecast to 2028,» February 2924. [Ηλεκτρονικό]. Available: https://www.marketsandmarkets.com/Market-Reports/automation-testing-market-113583451.html**

**[7] Smartbear, «2018 State of Testing Report | https://smartbear.com/resources/ebooks/state-of-testing-report-2018/».**

**[8] M. Honda, «The Importance of Visual Testing in Software Quality,» Katalon, Inc., 2018. [Ηλεκτρονικό]. Available: https://katalon.com/resources-center/blog/importance-of-visual-testing.**

**[9] Selenium, «Information about web elements v4.0,» [Ηλεκτρονικό]. Available: https://www.selenium.dev/documentation/webdriver/elements/information/.**

**[10] Katalon, «Visual Testing overview,» [Ηλεκτρονικό]. Available: https://docs.katalon.com/katalon-platform/analyze/analytics/visual-testing/visual-testing-overview.**

**[11] LambdaTest, «Getting started with LambdaTest Smart UI Figma-Web CLI,» [Ηλεκτρονικό]. Available: https://www.lambdatest.com/support/docs/smartui-cli-figma-web/.**

**[12] Selenium, «Locator Straegies v4.0,» [Ηλεκτρονικό]. Available: https://www.selenium.dev/documentation/webdriver/elements/locators/.**

**[13] Cypress, Inc, «Visual Testing in Cypress,» January 2025. [Ηλεκτρονικό]. Available: https://docs.cypress.io/app/tooling/visual-testing.**

**[14] SmartBear, «SmartBear VisualTest,» [Ηλεκτρονικό]. Available: https://smartbear.com/product/visualtest/**

**[15] Functionize, «Visual Testing,» [Ηλεκτρονικό]. Available: https://www.functionize.com/visual-testing.**

**[16] A. A. G. BS, H. Valluri, T. P. MD, A. C. MD, A. S. B. MD, J. C. D. MD, N. C. F. MD και A. R. P. MD, «Evaluating the Performance of ChatGPT-4o Vision Capabilities on Image-Based USMLE,», *medRxiv,* 2024. Available: https://doi.org/10.1101/2024.06.18.24309092.**

**[17] Y. Wu, X. Hu, Z. Fu, S. Zhou και J. Li, «GPT-4o: Visual perception performance of multimodal large language models in piglet activity understanding,», *ArXiv* , 2024. Available: https://arxiv.org/abs/2406.09781v1**

**[18] M. Moradi, K. Yan, D. Colwell και R. Asgari, «Artificial intelligence for context-aware visual change detection in software test automation,» *arXiv,* 2024. Available: https://doi.org/10.48550/arXiv.2405.00874**

**[19] P. W. T. X. T. H. Z. H. Z. Z. M. Z. S. Zechen Bai, «Hallucination of Multimodal Large Language Models: A Survey,», *arXiv,* 2024. Available: https://arxiv.org/abs/2404.18930v2.**

**[20] [Ηλεκτρονικό]. Available: https://platform.openai.com/docs/guides/text?api-mode=responses.**

**[21] «OpenAI API Pricing,» [Ηλεκτρονικό]. Available: https://openai.com/api/pricing/.**

# Παραρτήματα

## Π-1 Εγχειρίδιο Χρήστη

**Χρήση του εργαλείου:**

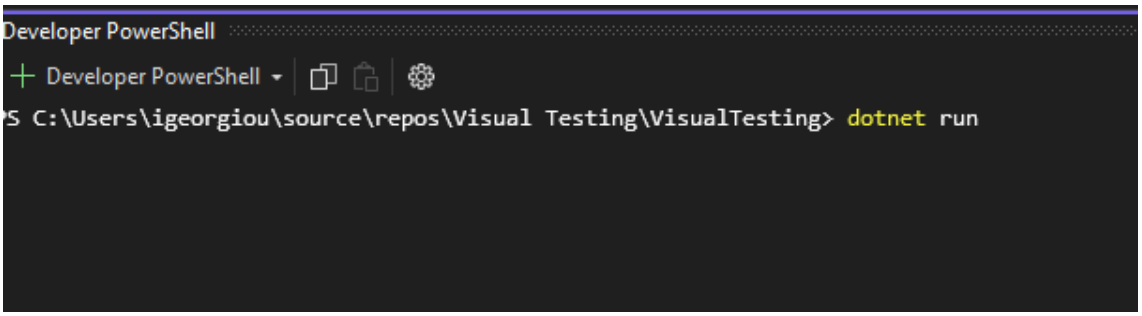
Εδώ γίνεται περιγραφή για το πως μπορεί να χρησιμοποιηθεί το εργαλείο. Στο Π-2 (Τεχνικό Εγχειρίδιο) θα γίνει σύντομη αναφορά για το πως μπορεί να εγκατασταθεί. Το εγχειρίδιο χρήστη είναι χωρισμένο με βάση τις κύριες λειτουργίες όπως αυτές αναφέρονται στα RSDs.

1. Εκτέλεση προγράμματος:

Αφού εγκατασταθεί το περιβάλλον σωστά όπως περιγράφεται στο (Π-2):

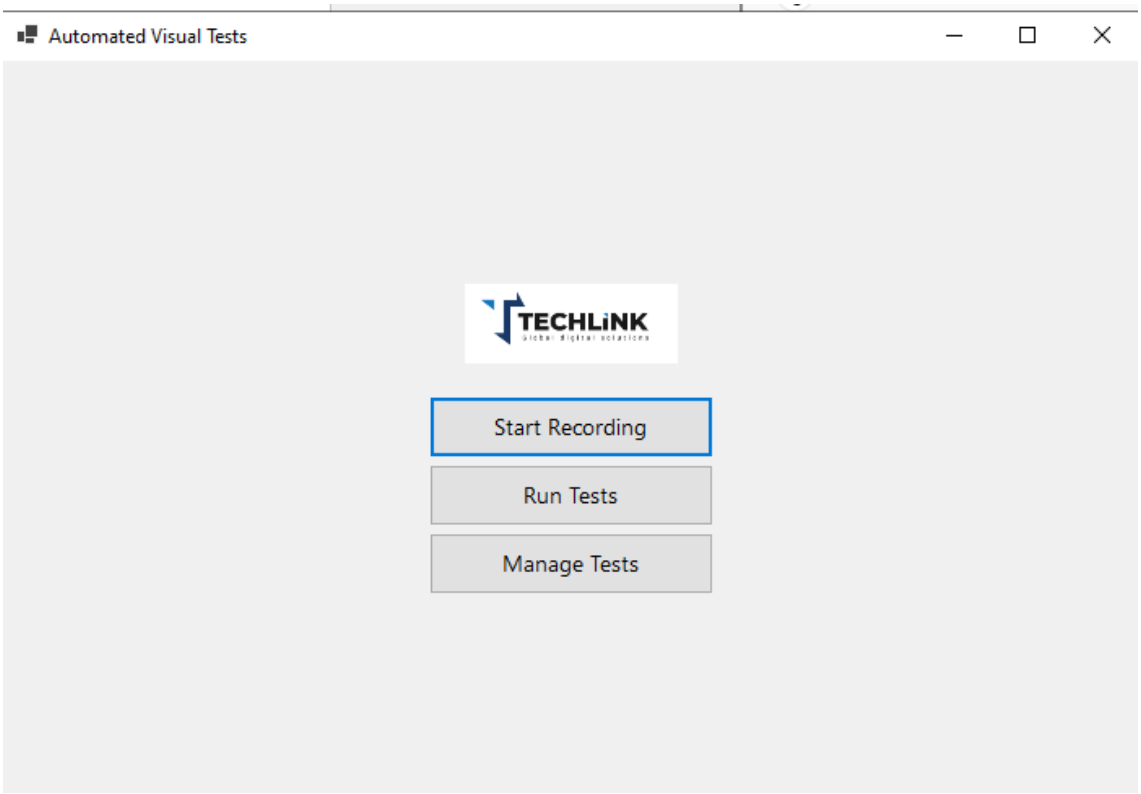
1.1 Πλοήγηση στην ρίζα του αρχείου του έργου

1.2 Πληκτρολόγηση εντολής εκτέλεσης dotnet εφαρμογής

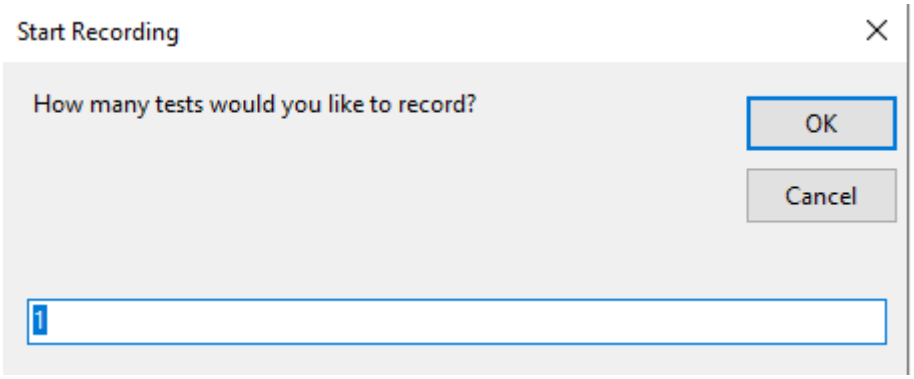



2. Εγγραφή δοκιμών:

2.1 Πάτησε το “Start Recording”

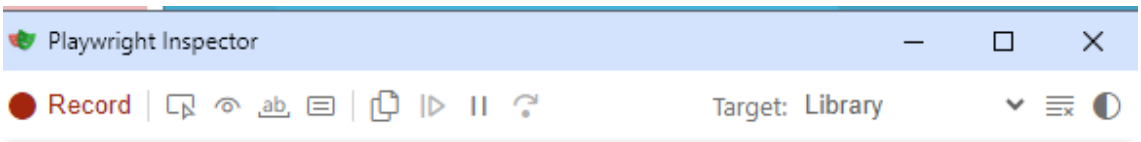



2.2 Εισαγωγή αρ. Δοκιμών




2.3 Κάνε όσα βήματα θέλεις μέχρι την οθόνη που είναι για έλεγχο

2.4 Στο τέλος κάθε δοκιμής πάτα το κόκκινο κουμπί του Playwright Codegen

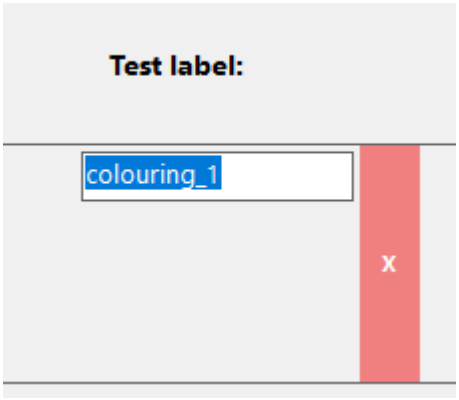



2.5 Επανάλαβε μέχρι να τελειώσουν οι δοκιμές

3. <u>Διαχείριση Δοκιμών:</u>

3.1 Πάτα στο Manage Tests

3.2 Για αλλαγή ονόματος, πάτα στο label και μετά πάρε αλλού τον δείκτη.




3.3 Για διαγραφή φωτογραφίας πάτα το κουμπί χ δίπλα από αυτήν

3.4 Για επιλογή άλλης πάτα πάνω στο γκρίζο κουτί ή σύρε μια φωτογραφία μέσα

4. Εκτέλεση δοκιμών:

4.1 Για την εκτέλεση δοκιμών πάτα το “Run Tests”

4.2 Περίμενε μέχρι να εμφανιστεί μήνυμα ολοκλήρωσης.

5. Ανάγνωση Αποτελεσμάτων:

5.1 Εάν έχουν παραχθεί αποτελέσματα, θα βρίσκονται στο TestResults.html κάτω από τον ίδιο φάκελο με τον κώδικα της εφαρμογής

5.2 Θα είναι αυτής της δομής. Συνολικός αριθμός δοκιμών, συνολικά αποτυχημένα και συνολικά που πέρασαν. Στον πίνακα θα εμφανίζονται τα στοιχεία ως εξής. Κόκκινο – Αποτυχία. Πράσινο – Επιτυχία.,

**Visual Test Results**

Total number of tests: 100

**Total failed**: 41

**Total passed**: 59

| File | Status | Label | Design | Screenshot | Message |
|---|---|---|---|---|---|
| 0.png | Passed | colouring_1 | Design | Screenshot | |
| 1.png | Passed | colouring_2 | Design | Screenshot | |
| 2.png | Failed | colouring_3 | Design | Screenshot | - The "Επεξεργασία" links in the second image are displayed in green, whereas in the first image they are blue. |
| 3.png | Passed | colouring_4 | Design | Screenshot | |
| 4.png | Passed | colouring_5 | Design | Screenshot | |

## Π-2 Τεχνικό Εγχειρίδιο

Αυτό το εγχειρίδιο απευθύνεται σε προγραμματιστές και τεχνικούς που χρειάζονται να εγκαταστήσουν, να ρυθμίσουν ή να συντηρήσουν το σύστημα αυτόματου οπτικού ελέγχου που βασίζεται σε C#, Playwright και OpenAI.

1. Ελάχιστες Απαιτήσεις Συστήματος

| Απαίτηση | Ελάχιστο |
|---|---|
| Λειτουργικό Σύστημα | Windows 10 |
| Μνήμη RAM | 4GB |
| Ελεύθερος Χώρος Δίσκου | 25GB |
| Περιηγητής | Google Chrome (latest version) |
| Σύνδεση Internet | Για την εκτέλεση και την εγγραφή των δοκιμών |

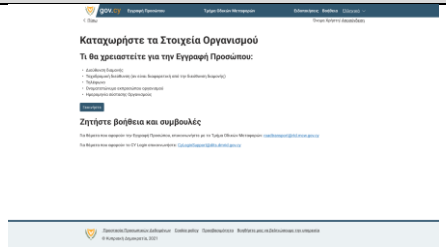
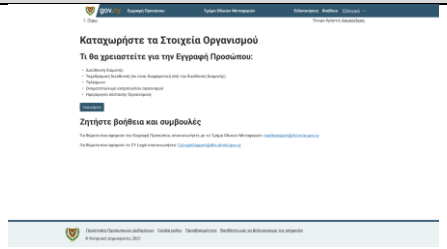
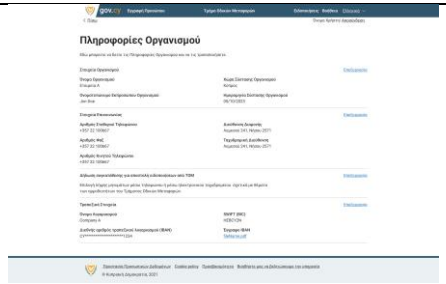
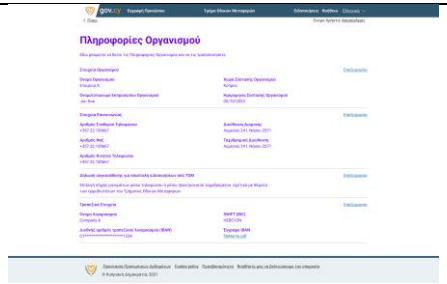
2. **Προαπαιτούμενα εργαλεία**
   - Visual Studio 2022 (.NET Desktop Development, ASP.NET and web development)
   - .NET 8.0
   - Playwright for .NET
   - nUnit Test Adapter
   - Google Chrome and Chromium
   - Access to Azure Repos of Project
   - OpenAI Api Key

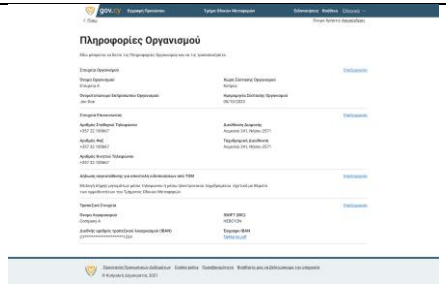
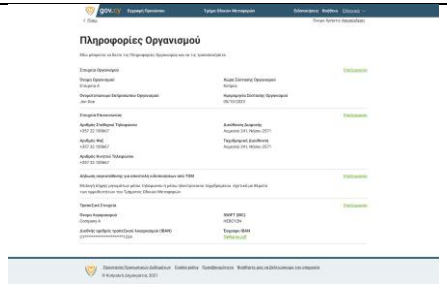
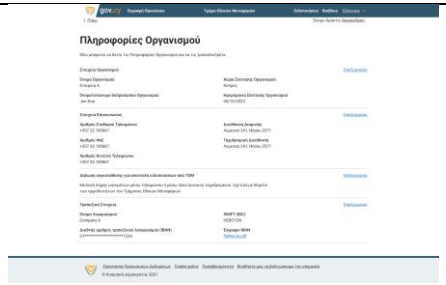
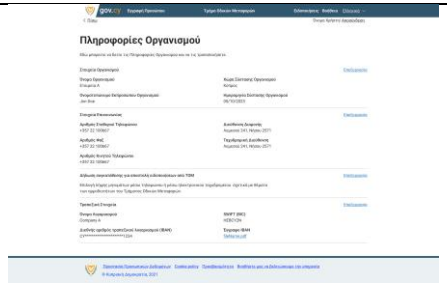
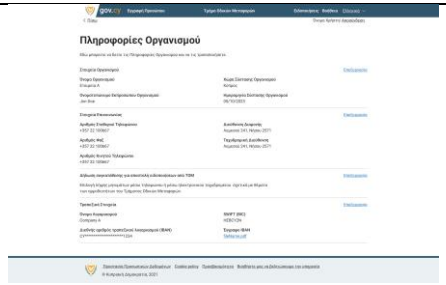
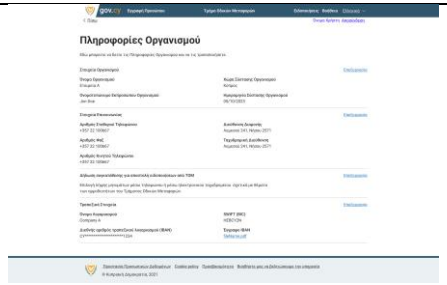
3. Βήματα εγκατάστασης
   - **a.** Σύνδεση στον λογαριασμό Azure DevOps
   - **b.** Κλωνοποίηση αποθετηρίου κώδικα
   - **c.** Άνοιγμα .sln αρχείου
   - **d.** Εγκατάσταση Playwright:
     - **i. dotnet add package Microsoft.Playwright**
     - **ii. playwright install**

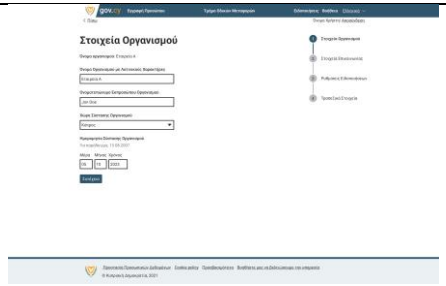
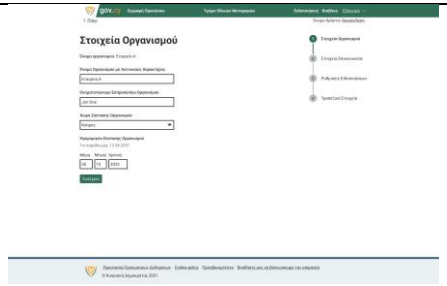
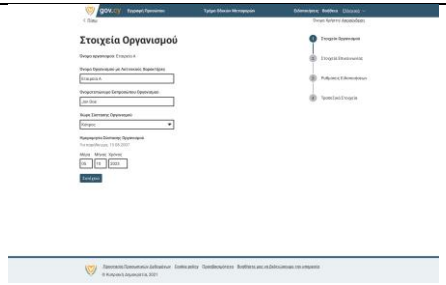
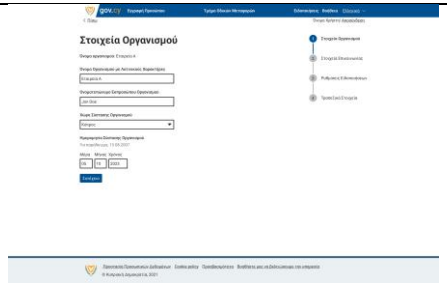
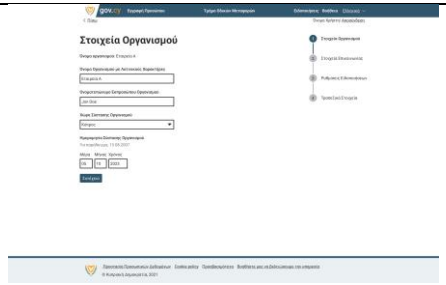
e. Ρύθμιση OpenAI Key στο κατάλληλο σημείο (κατά προτίμηση)
f. Εκτέλεση μέσω dotnet run

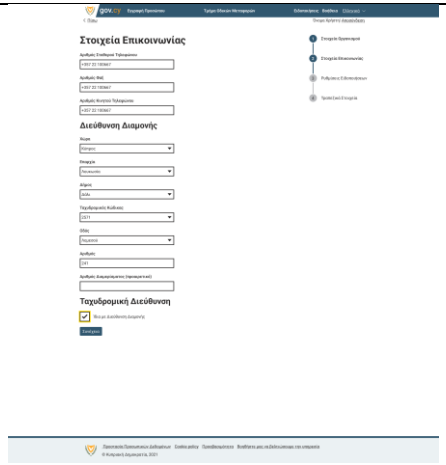
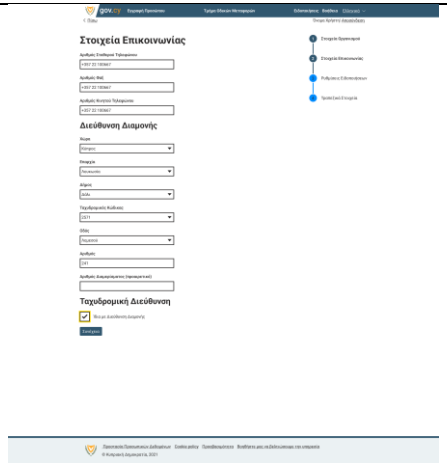
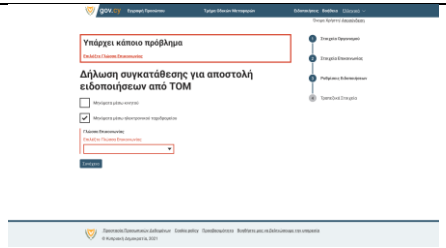
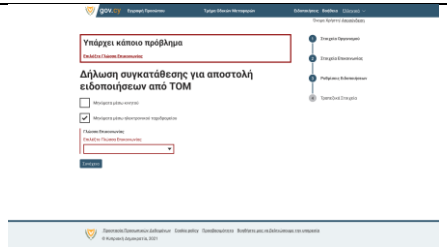
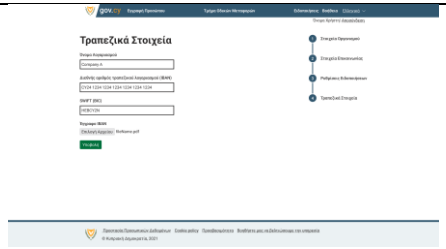
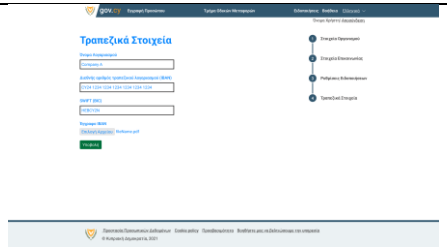
## Π-3 Δείγματα

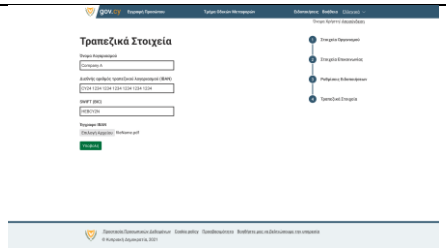
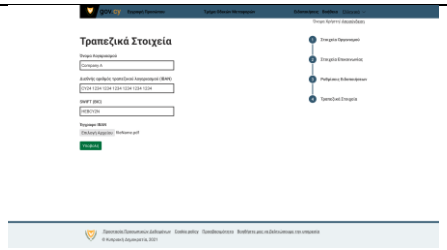
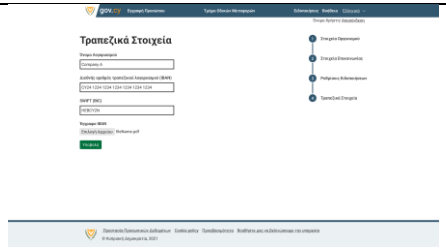
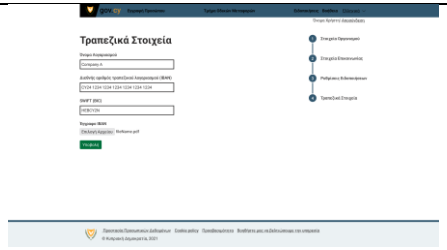
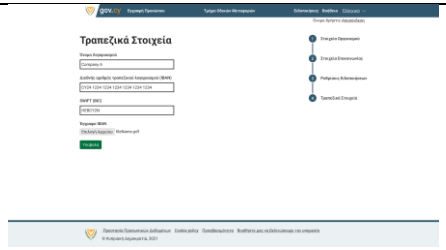
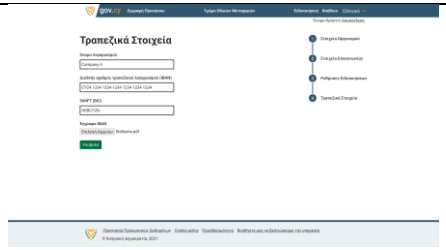
Πιο κάτω εμφανίζονται τα δείγματα με τα οποία έγινε η αξιολόγηση της εφαρμογής.

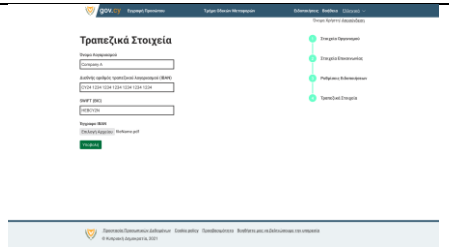
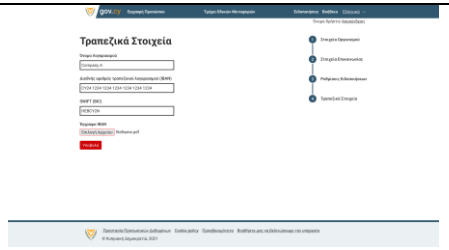
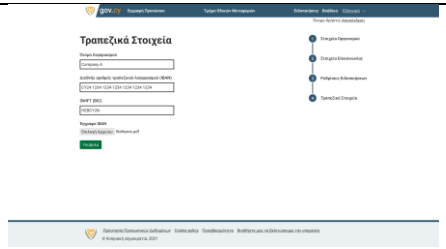
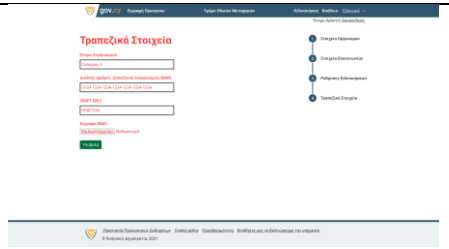
| Ονομασία | Σχέδιο Βάσης | Στιγμιότυπο |
|---|---|---|
| Colouring_1 |  |  |
| Διαφορές | 1. The colouring of the Cyprus emblem and the .cy is a darker orange | |
| Colouring_2 |  |  |
| Διαφορές | 1. Purple text and not black text | |

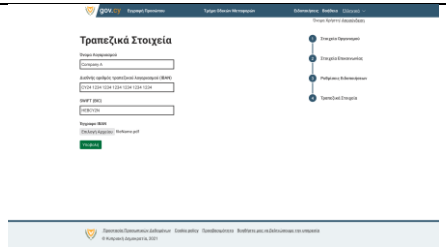
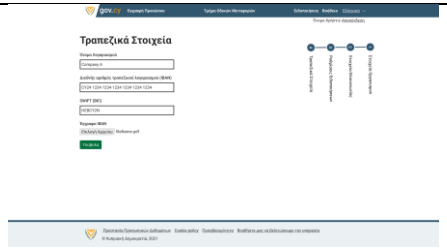
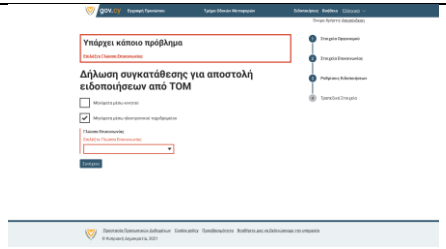
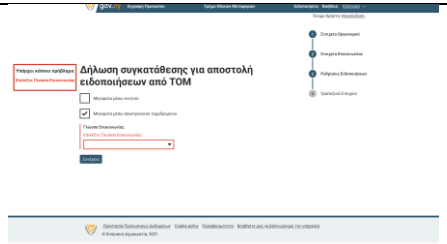
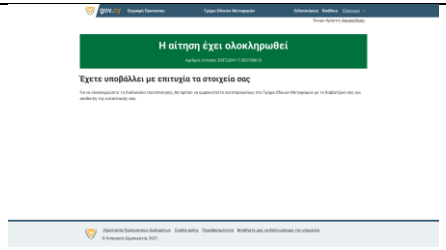
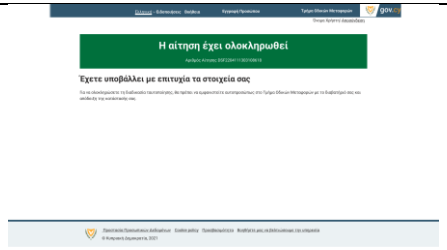
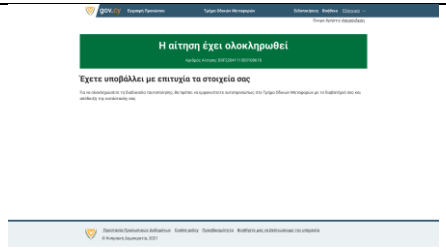
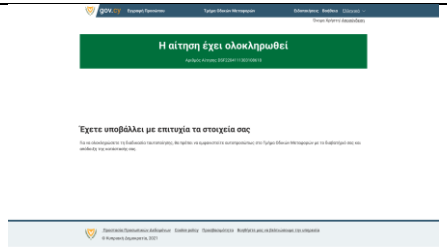
| | | |
|---|---|---|
| Colouring_ 3 |  |  |
| Διαφορές | 1. Green links vs blue links | |
| Colouring_ 4 |  |  |
| Διαφορές | 1. Blue links and text on footer | |
| Colouring_ 5 |  |  |
| Διαφορές | 1. Blue links on user name and log out | |

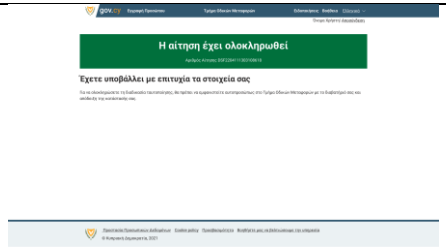
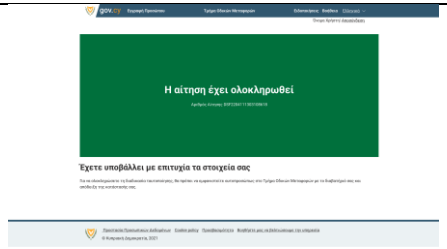
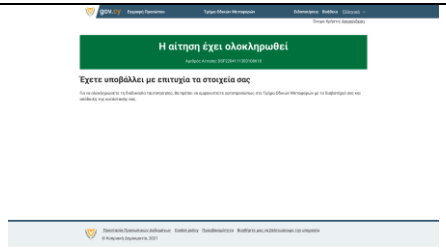
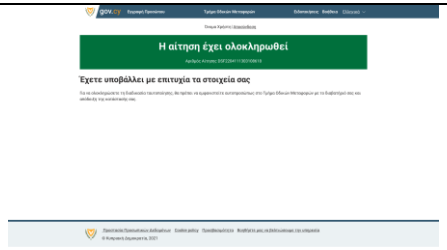
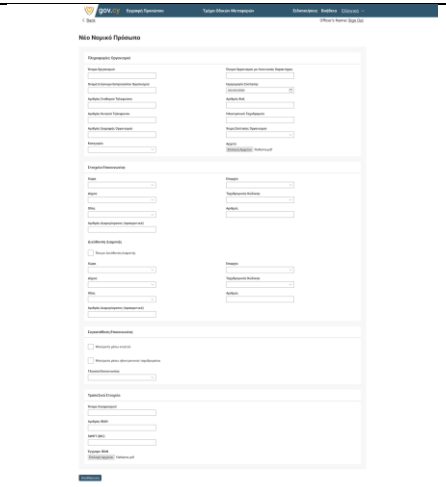
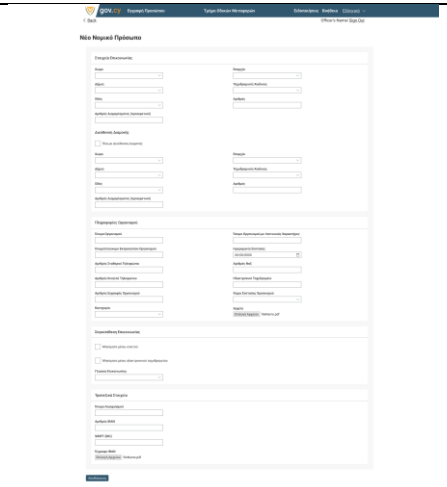
| | | |
|---|---|---|
| Colouring_ 6 |  |  |
| Διαφορές | 1. Different header colour 2. Different button colour 3. Different step colour | |
| Colouring_ 7 |  |  |
| Διαφορές | 1. Different hue of blue header 2. Different hue of blue button3. Different hue of blue continue button | |
| Colouring_ 8 |  |  |
| Διαφορές | 1. Different background colour | |

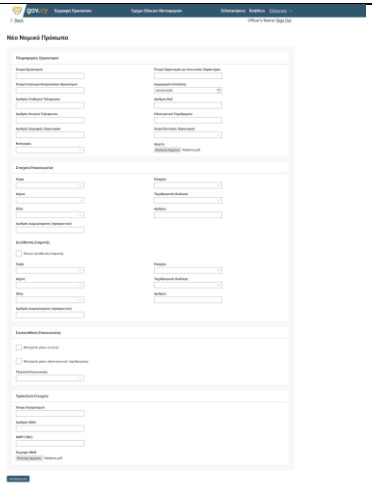
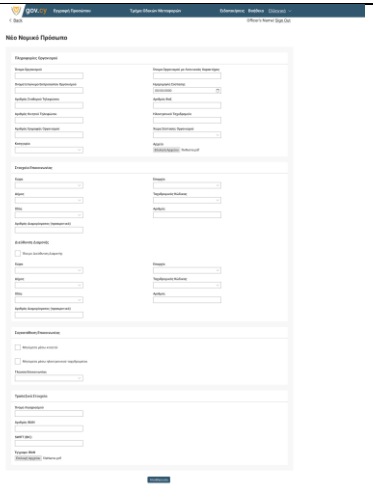
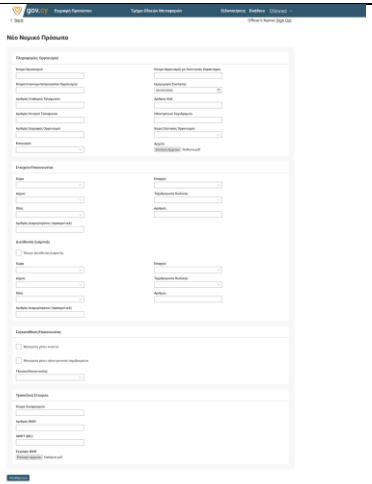
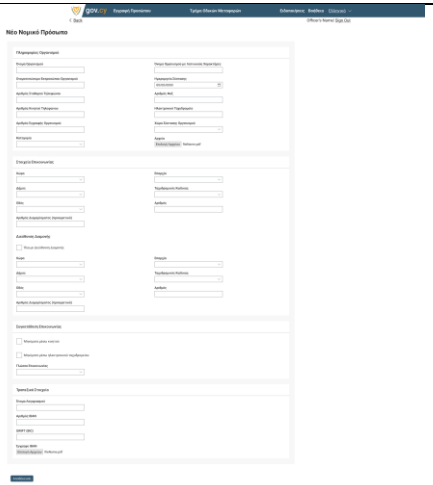
| | | |
|---|---|---|
| Colouring_ 9 |  |  |
| Διαφορές | 1. Different stepper colour | |
| Colouring_ 10 |  |  |
| Διαφορές | 1. Different hue of red in error message | |
| Colouring_ 11 |  |  |
| Διαφορές | 1. Different text colour (blue vs black) | |

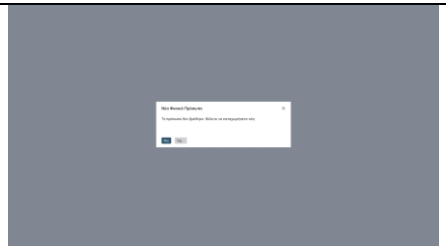
| | | |
|---|---|---|
| Colouring_ 12 |  |  |
| Διαφορές | 1. Different colouring for header text and background | |
| Colouring_ 13 |  |  |
| Διαφορές | 1. Different colouring in the text of text stepper | |
| Colouring_ 14 |  |  |
| Διαφορές | 1. gray vs black in text colour | |
| Colouring_ 15 |  |  |

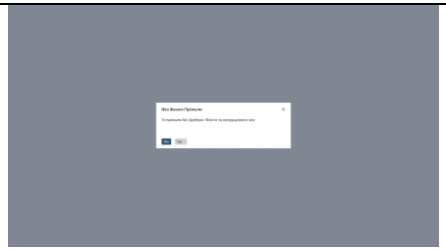
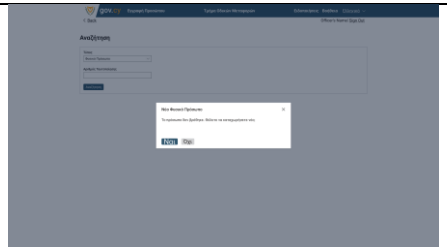
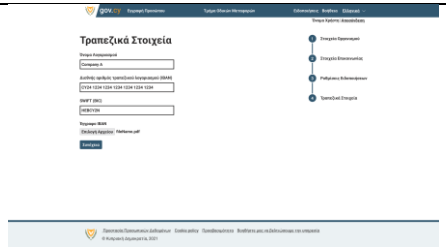
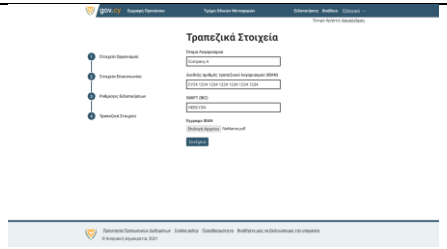
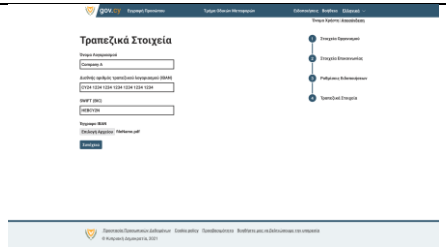
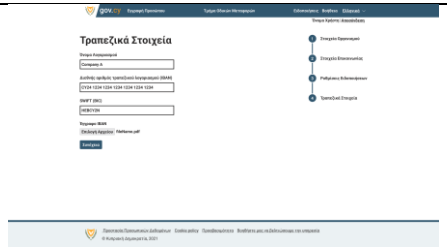
| | | |
|---|---|---|
| Διαφορές | 1. gray field background vs white | |
| Colouring_ 16 |  |  |
| Διαφορές | 1. light green stepper circles vs blue stepper circles | |
| Colouring_ 17 |  |  |
| Διαφορές | 1. Submit red colour 2. File option red colour | |
| Colouring_ 18 |  |  |
| Διαφορές | 1. Different title colour (red) 2. Different field title colours (red)3. Different in field text colours (red) | |

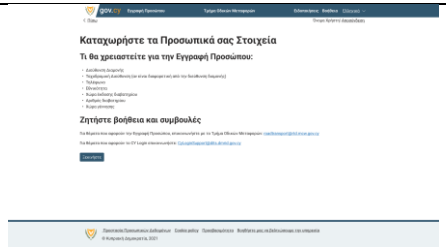
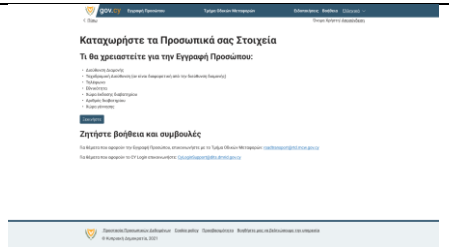
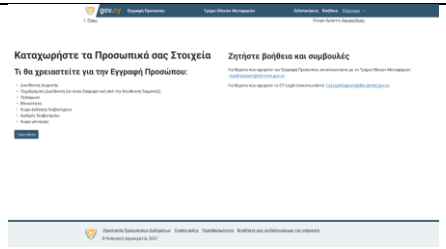
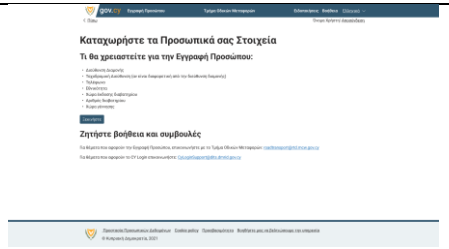
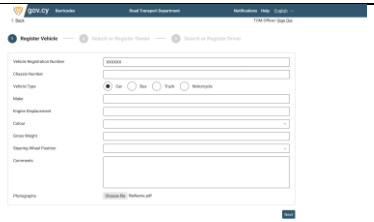
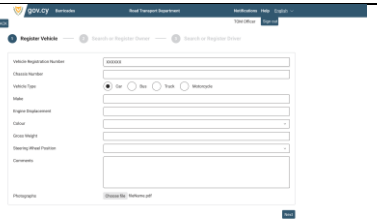
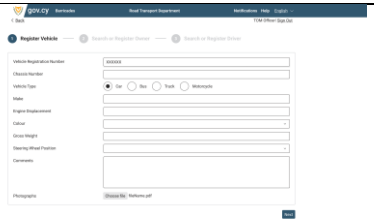
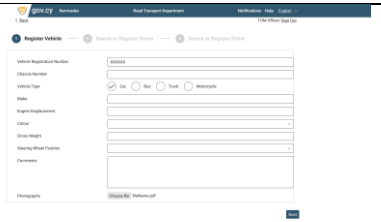
| | | |
|---|---|---|
| Layout_1 |  |  |
| Διαφορές | 1. Stepper design is different - fliped. | |
| Layout_2 |  |  |
| Διαφορές | 1. Error message on the left | |
| Layout_3 |  |  |
| Διαφορές | 1.gov.cy logo is on the right instead of the left | |
| Layout_4 |  |  |

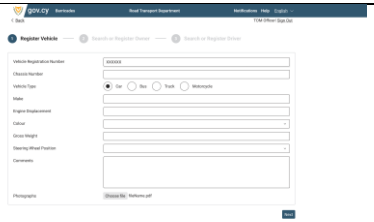
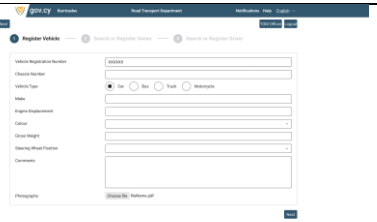
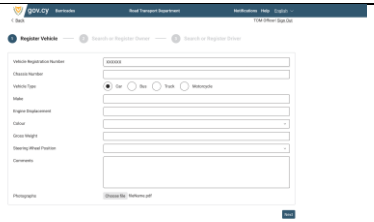
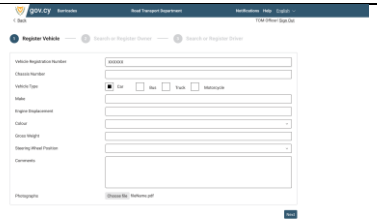
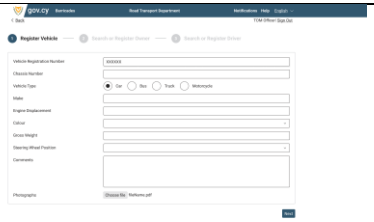
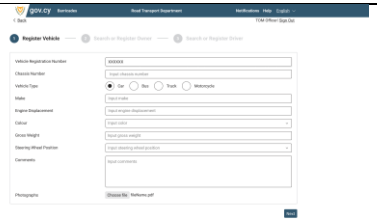
| | | |
|---|---|---|
| Διαφορές | 1. The success information message is further on the bottom, not directly below the notification message | |
| Layout_5 |  |  |
| Διαφορές | 1. Larger in heigth notification nmessage 2. Smaller spacing between notification message and message | |
| Layout_6 |  |  |
| Διαφορές | 1. User info and log out are centered | |
| Layout_7 |  |  |
| Διαφορές | 1. Form list elements are positioned differently | |

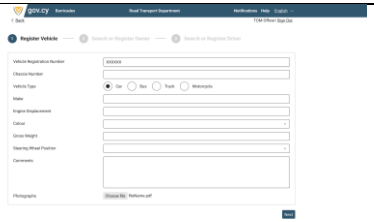
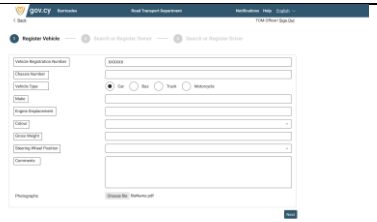
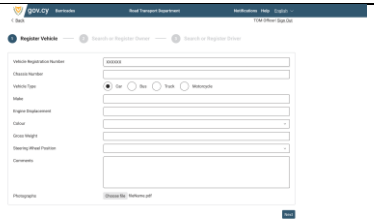
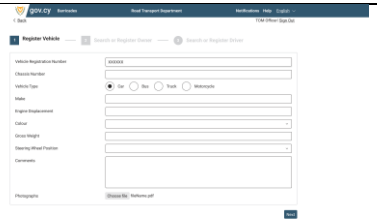
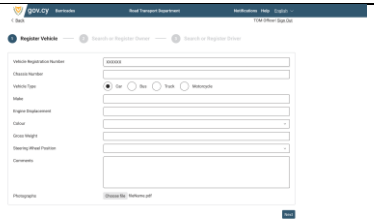
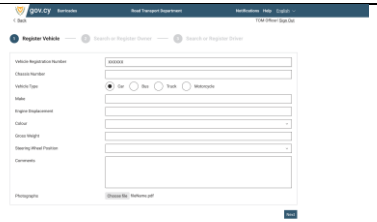
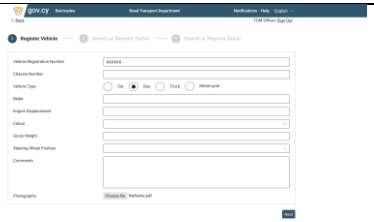
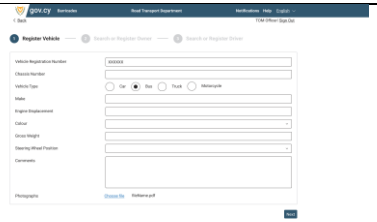
| | | |
|---|---|---|
| Layout_8 |  |  |
| Διαφορές | 1. Button is centered | |
| Layout_9 |  |  |
| Διαφορές | 1. Form is aligned on the left not centered | |

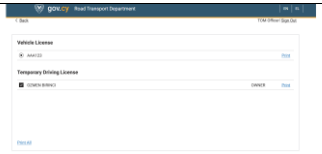
| | | |
|---|---|---|
| Layout_10 |  |  |
| Διαφορές | 1. Search button is a bit larger 2. Search button is spaced differently in context box | |
| Layout_11 |  |  |
| Διαφορές | 1. Search title is aligned on the right not centered | |
| Layout_12 |  |  |
| Διαφορές | 1. Search button is on the top | |
| Layout_13 |  |  |

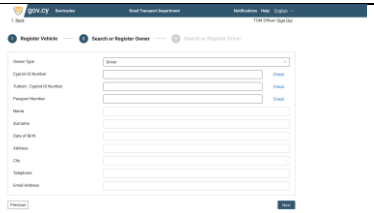
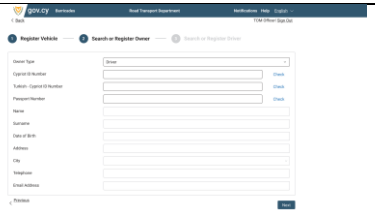
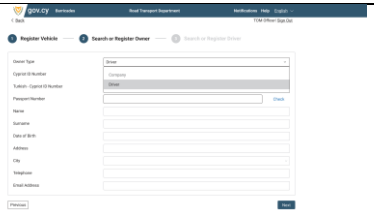
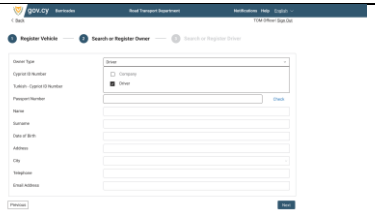
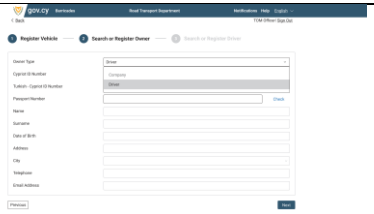
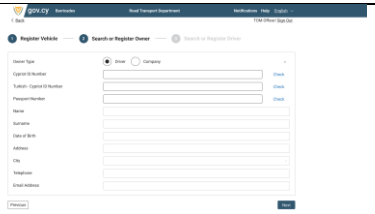
| | | |
|---|---|---|
| Διαφορές | 1. Info message is cenetered | |
| Layout_14 |  |  |
| Διαφορές | 1. Size is different 2. Style is different | |
| Layout_15 |  |  |
| Διαφορές | 1. Stepper is on the right not the left | |
| Layout_16 |  |  |
| Διαφορές | 1. Bold letters with less space | |

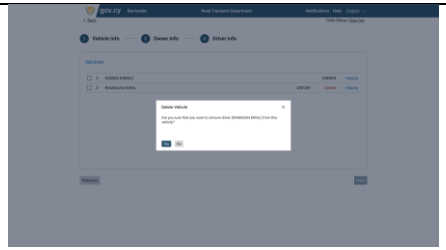
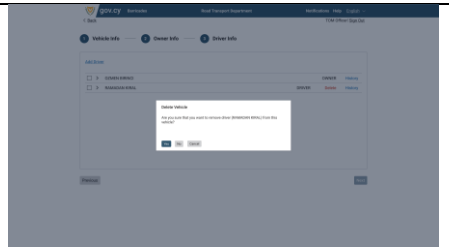
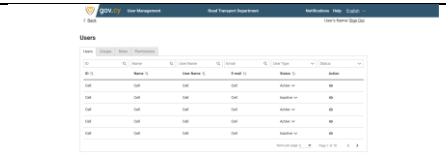
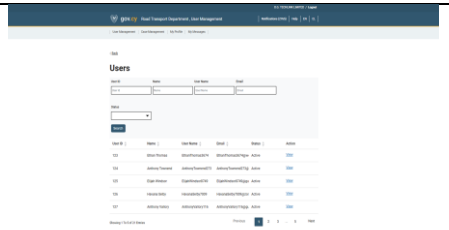
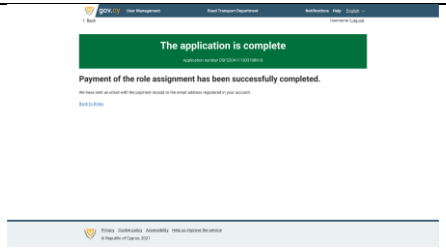
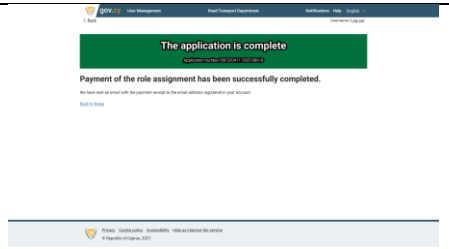
| | | |
|---|---|---|
| Layout_17 |  |  |
| Διαφορές | 1. Begin button is placed elsewhere | |
| Layout_18 |  |  |
| Διαφορές | 1. Infos are on the left not in the middle | |
| Design_1 |  |  |
| Διαφορές | 1. Different back button 2. Different sign out design | |
| Design_2 |  |  |

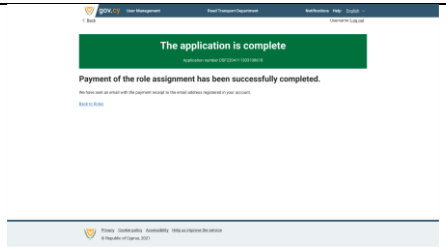
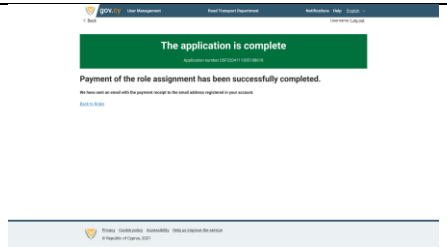
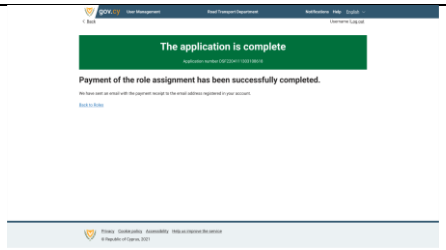
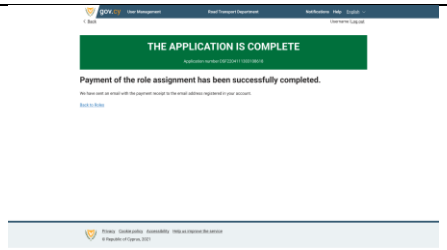
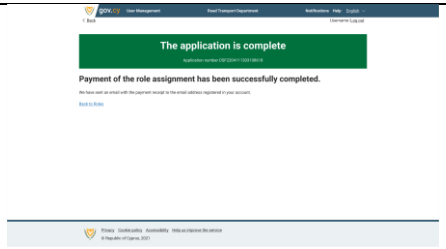
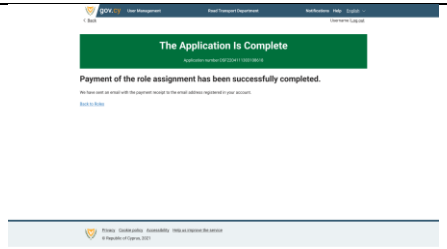
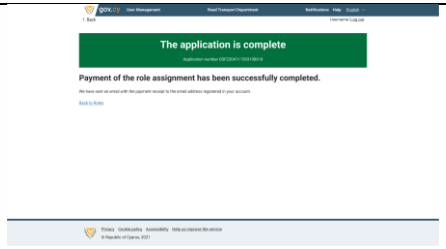
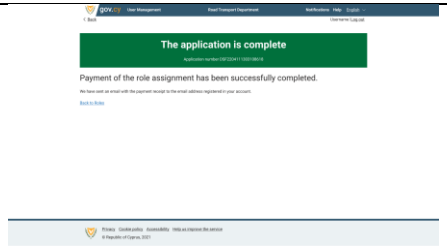
| | | |
|---|---|---|
| Διαφορές | 1. Check instead of radio button | |
| Design_3 |  |  |
| Διαφορές | 1. Back is replaced with next button 2. Signout is replaced with logout 3. User info changed style | |
| Design_4 |  |  |
| Διαφορές | 1. Rectangular vs circucal buttons 2. Black circle vs black square | |
| Design_5 |  |  |
| Διαφορές | 1. Input comments | |

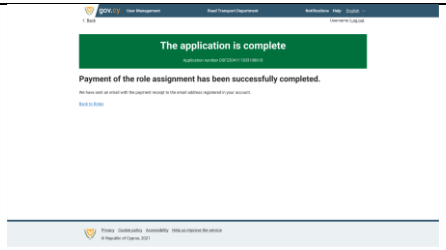
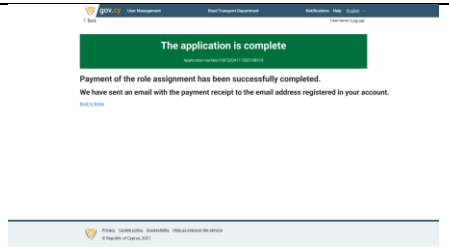
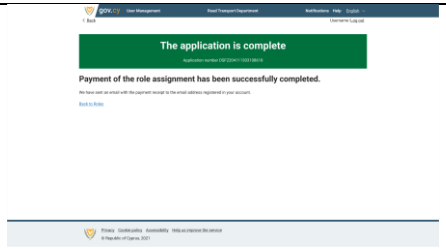
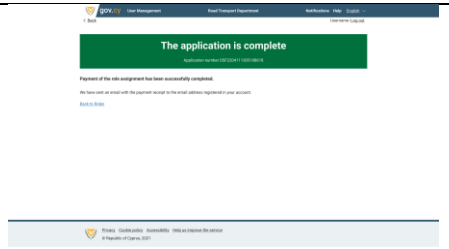
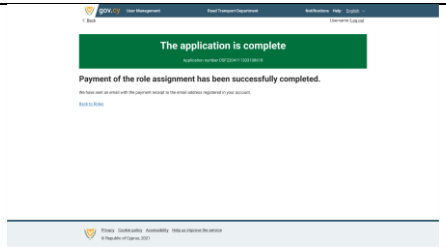
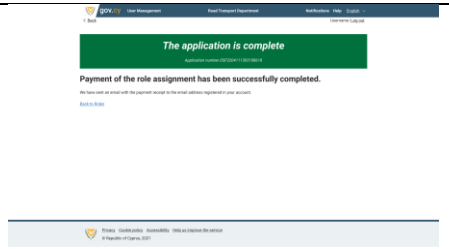
| | | |
|---|---|---|
| Design_6 |  |  |
| Διαφορές | 1. Squares around labels | |
| Design_7 |  |  |
| Διαφορές | 1. Square stepper | |
| Design_8 |  |  |
| Διαφορές | 1. No rounded corners | |
| Design_9 |  |  |

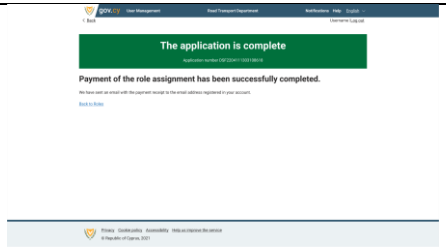
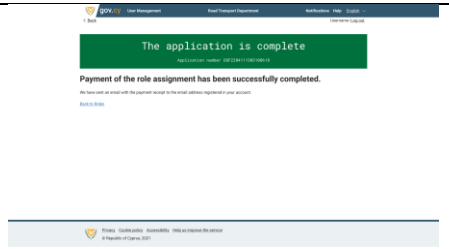
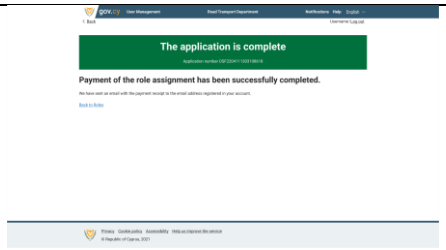
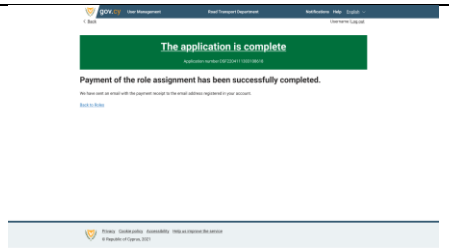
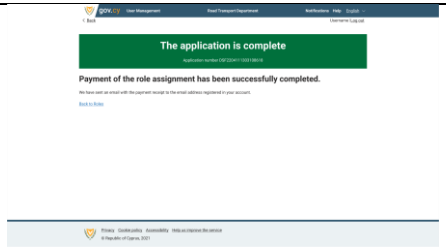
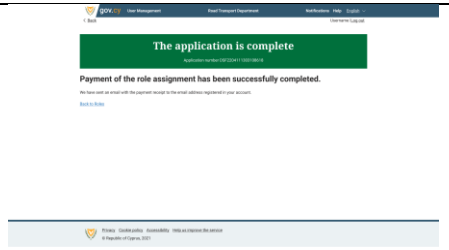
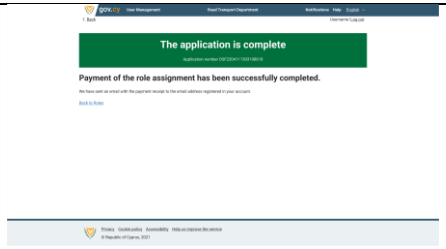
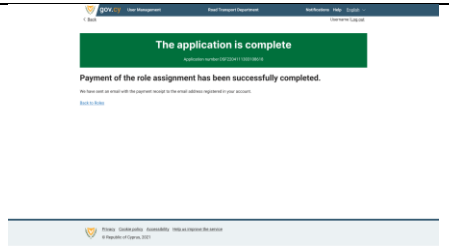
| | | |
|---|---|---|
| Διαφορές | 1. Choose file link not button | |
| Design_10 |  |  |
| Διαφορές | 1. Menu instead of tabs | |
| Design_11 |  |  |
| Διαφορές | 1. Different menu item 2. Yellow border vs no yellow border 3. Language toggle vs language dropdown 4. Different customized font in banner | |
| Design_12 |  |  |
| Διαφορές | 1. Check buttons vs check links | |

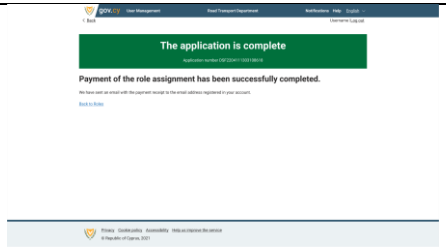
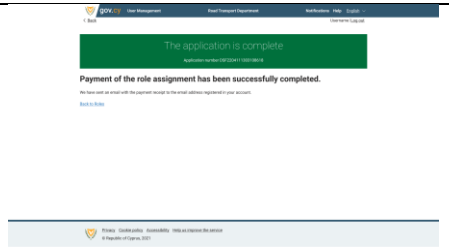
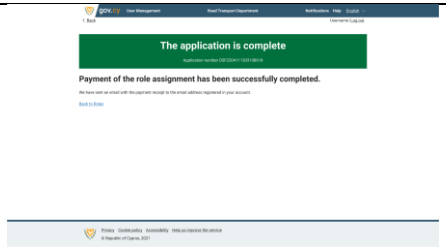
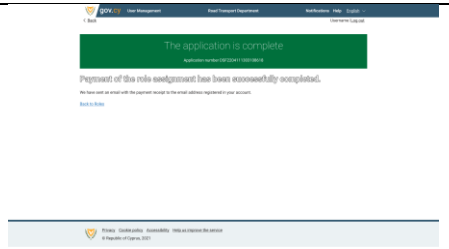
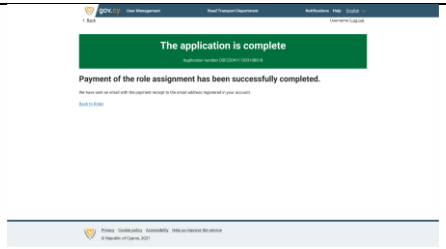
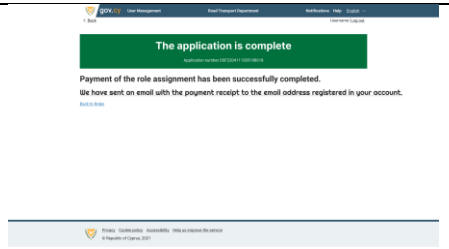
| | | |
|---|---|---|
| Design_13 |  |  |
| Διαφορές | 1. Previous link instead of previous button | |
| Design_14 |  |  |
| Διαφορές | 1. Checkbox instead of dropdown | |
| Design_15 |  |  |
| Διαφορές | 1. Dropdown vs radio buttons | |
| Design_16 |  |  |

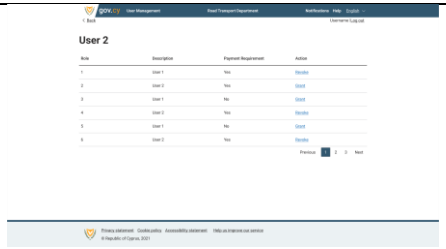
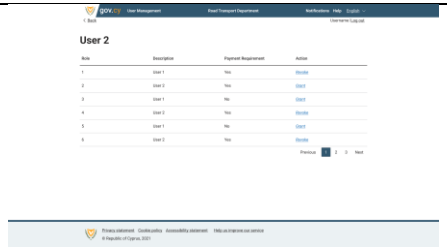
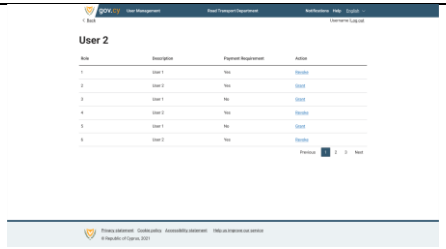
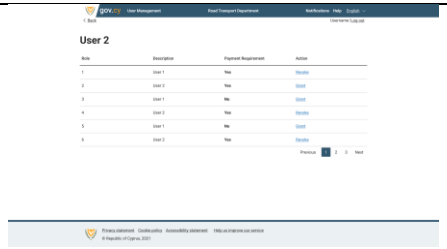
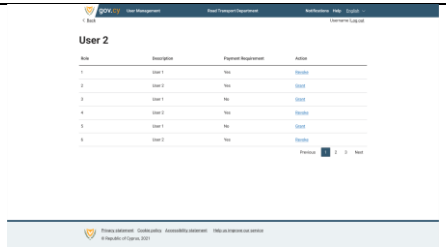
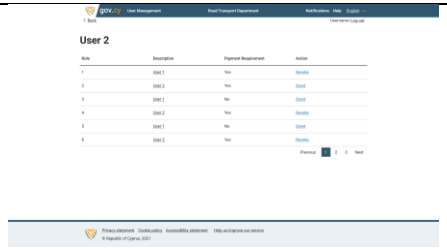
| | | |
|---|---|---|
| Διαφορές | 1. Print button vs print link 2. Print all button vs print all link 3. Different button text design | |
| Design_17 |  |  |
| Διαφορές | 1. Cancel button 2. No x button | |
| Design_18 |  |  |
| Διαφορές | 1. Different stylization of text contents 2. Different search button 3. Different pagination controls 5. View vs eye button | |
| fonts_1 |   |   |
| Διαφορές | 1. Outline vs no outline for header 2. Outline vs no outline for app text | |

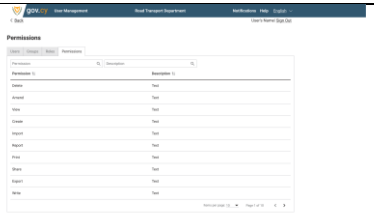
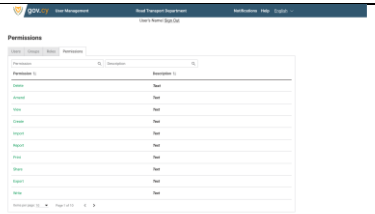
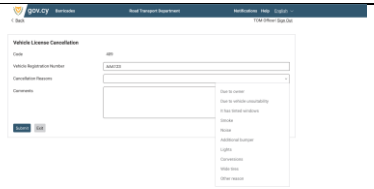
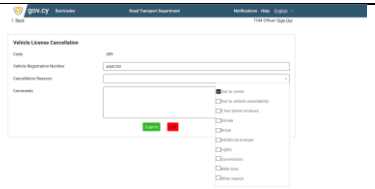
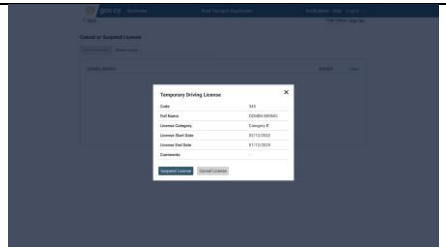
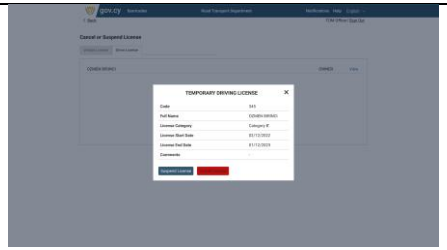
| | | |
|---|---|---|
| Fonts_2 |  |  |
| Διαφορές | 1. bold info message | |
| Fonts_3 |  |  |
| Διαφορές | 1. Capital and larger text | |
| Fonts_4 |  |  |
| Διαφορές | 1. Title Case | |
| Fonts_5 |  |  |

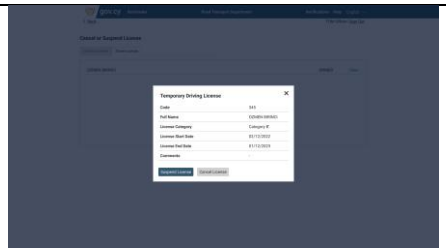
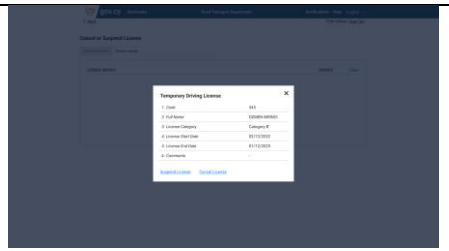
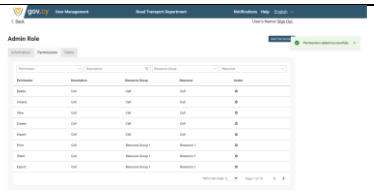
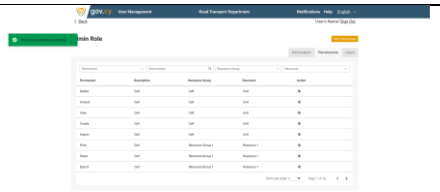
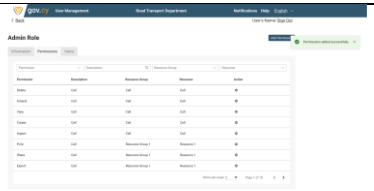
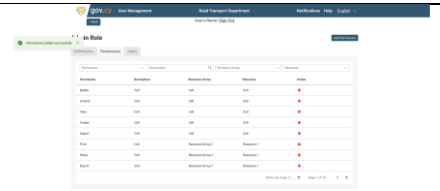
| | | |
|---|---|---|
| Διαφορές | 1. no bold text | |
| Fonts_6 |  |  |
| Διαφορές | 1. everything is bold and larger in the info message | |
| Fonts_7 |  |  |
| Διαφορές | 1. smaller text in info message | |
| Fonts_8 |  |  |
| Διαφορές | 1. italics text in application | |

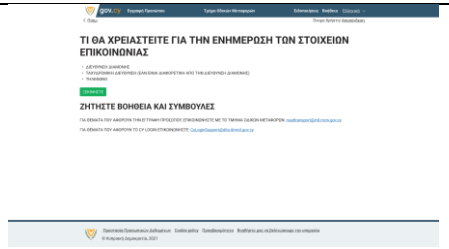
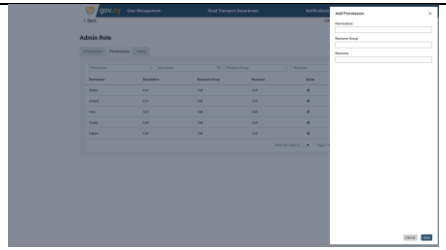
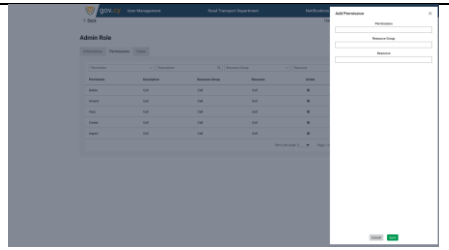
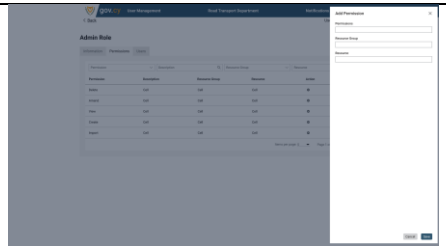
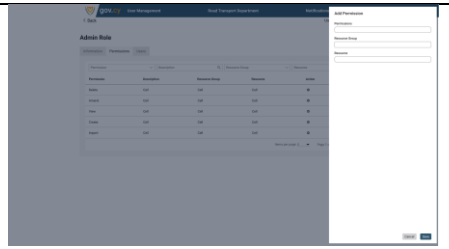
| | | |
|---|---|---|
| Fonts_9 |  |  |
| Διαφορές | 1. difference in font | |
| Fonts_10 |  |  |
| Διαφορές | 1. underlined text | |
| Fonts_11 |  |  |
| Διαφορές | 1. different text font | |
| Fonts_12 |  |  |

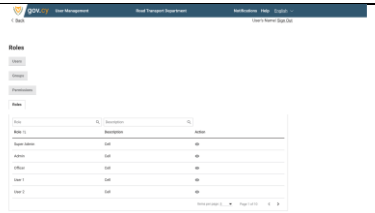
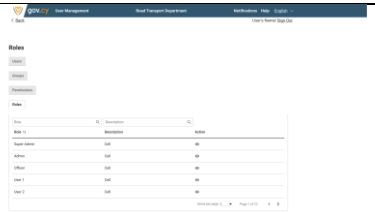
| | | |
|---|---|---|
| Διαφορές | 1. different text font | |
| Fonts_13 |  |  |
| Διαφορές | 1. Different font text | |
| Fonts_14 |  |  |
| Διαφορές | 1. Different font text for message as well 2. Different application text as well | |
| Fonts_15 |  |  |
| Διαφορές | 1. different font style 2. larger letters 3. underlined | |

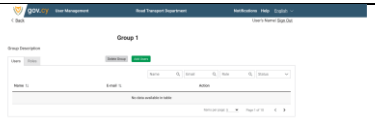
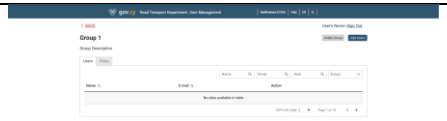
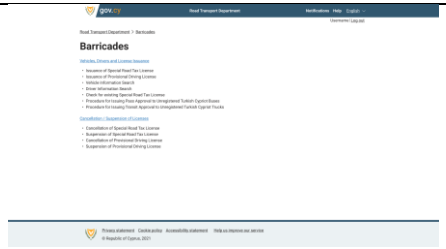
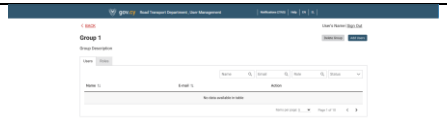
| | | |
|---|---|---|
| Fonts_16 |  |  |
| Διαφορές | 1. italics vs no italics | |
| Fonts_17 |  |  |
| Διαφορές | 1. bold vs no bold | |
| Fonts_18 |  |  |
| Διαφορές | 1. underline vs no underline | |

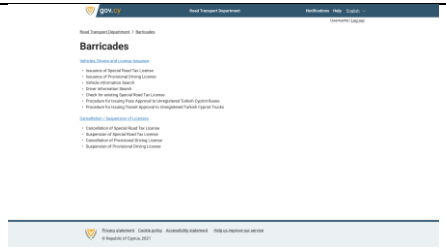
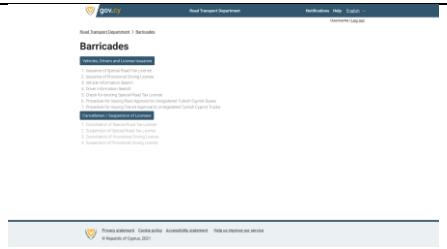
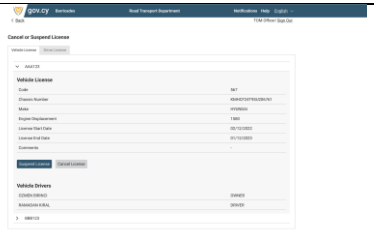
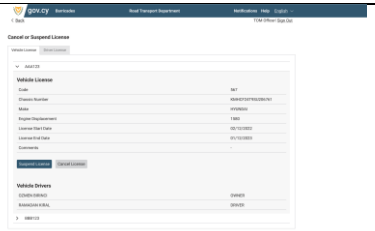
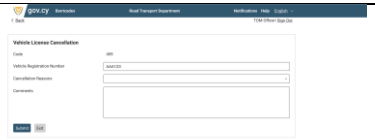
| | | |
|---|---|---|
| mix_1 |  |  |
| Διαφορές | 1. Green text vs not green text 2. Italic vs no italic 3. Different pagination control alignment | |
| Mix_2 |  |  |
| Διαφορές | 1. Submit is different 2. exit is different 3. checkbox vs dropdown | |
| Mix_3 |  |  |
| Διαφορές | 1. Capital vs not capital 2. Different cancel button | |

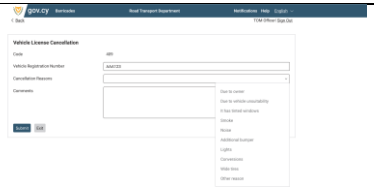
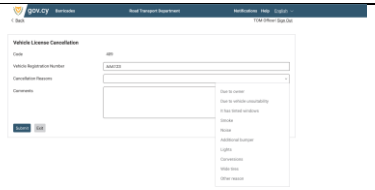
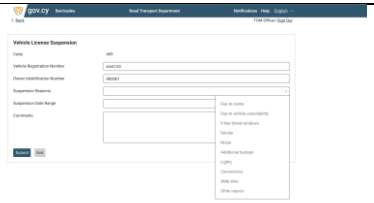
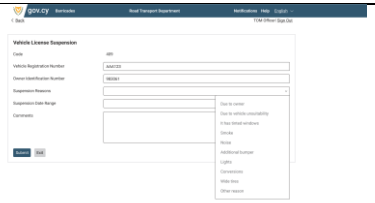
| | | |
|---|---|---|
| Mix_4 |  |  |
| Διαφορές | 1. italics vs not italics 2. italics vs no italics 3. links vs buttons | |
| Mix_5 |  |  |
| Διαφορές | 1. Different add permission colour 2. Different notification of success 3. different position of success | |
| Mix_6 |  |  |
| Διαφορές | 1. different position of permission message 2. different action buttons | |
| Mix_7 |  |  |

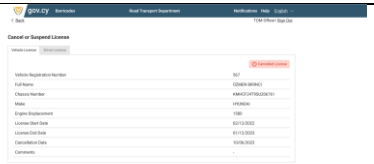
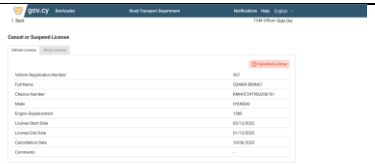
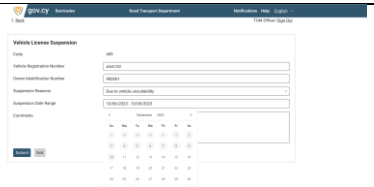
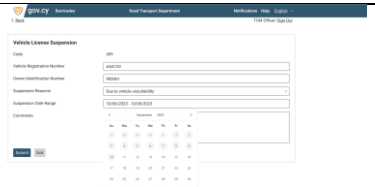
| | | |
|---|---|---|
| Διαφορές | 1. different button placement 2. bullet points vs numbering | |
| Mix_8 |  |  |
| Διαφορές | 1. Font difference 2. Button design difference | |
| Mix_9 |  |  |
| Διαφορές | 1. Save is different 2. Different text alignment | |
| Mix_10 |  |  |
| Διαφορές | 1. Rounded text field 2. no x button | |

| | | |
|---|---|---|
| Mix_11 |  |  |
| Διαφορές | 1. Tabs vs buttons 2. Tab placement | |
| Mix_12 |  |  |
| Διαφορές | 1. Different color for eyes 2. Italics description | |
| Mix_13 |  |  |
| Διαφορές | 1. Different in pagination design 2. Different in pagination placement | |
| Mix_14 |  |  |

| | | |
|---|---|---|
| Διαφορές | 1. Background colour is different 2. Text colouring is different 3. different font (capital) | |
| Mix_15 |  |  |
| Διαφορές | 1. Add users is different design 2. Different button placement | |
| Mix_16 |  |  |
| Διαφορές | 1. different back button 2. different notifications placement 3. different language placement 4. different header alignment | |
| Mix_17 |  |  |
| Διαφορές | 1. different font of barricade 2. different capitalization and font 3. centered aligned | |

| | | |
|---|---|---|
| Mix_18 |  |  |
| Διαφορές | 1. different stylization of texts 2. grayed out 3. numbering vs bullet point 4. different space alignment | |
| Same_1 |  |  |
| Διαφορές | | |
| Same_2 |  |  |
| Διαφορές | | |
| Same_3 |  |  |
| Διαφορές | | |

| | | |
|---|---|---|
| Same_4 |  |  |
| Διαφορές | | |
| Same_5 |  |  |
| Διαφορές | | |
| Same_6 |  |  |
| Διαφορές | | |
| Same_7 |  |  |
| Διαφορές | | |

| | | |
|---|---|---|
| Same_8 |  |  |
| Διαφορές | | |
| Same_9 |  |  |
| Διαφορές | | |
| Same_10 |  |  |
| Διαφορές | | |