­

**2Designing Online Personal Environment**

**Use D.O.P.E. to become a finer designer**

Σχεδίαση Συστήματος

Del.3.1

**Version 1.0**

**Χρήστος Παπακωνσταντίνου** [**papachri@ece.auth.gr**](mailto:papachri@ece.auth.gr)

**Αντώνης Μαυρομανώλης** [**antomavr@ece.auth.gr**](mailto:antomavr@ece.auth.gr)

**Γιώργος Μιχαηλίδης geormich@auth.gr**

**Θωμάς Πλιάκης** [**tpliakis@ece.auth.gr**](mailto:tpliakis@ece.auth.gr)

**28/05/2019­­­­­**

Ιστορικό Αλλαγών

|  |  |  |  |
| --- | --- | --- | --- |
| **Όνομα** | **Ημ/νία** | **Περιγραφή Αλλαγής** | **Εκδ.** |
| Α. Συμεωνίδης | 29/05/2009 | Δημιουργία Εγγράφου  Προσαρμογή του ESA software engineering standards guidelines (1991) και του εγγράφου SDD document, από τους Bruegge και Dutoit (2004). | 0.1 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Μέλη της Ομάδας Ανάπτυξης

|  |  |  |
| --- | --- | --- |
| **Όνομα** | **OA** | **Email** |
| Αντώνης Μαυρομανώλης | 40 | [antomavr@ece.auth.gr](mailto:antomavr@ece.auth.gr) |
| Θωμάς Πλιάκης | 40 | [tpliakis@ece.auth.gr](mailto:tpliakis@ece.auth.gr) |
| Χρήστος Παπακωνσταντίνου | 40 | [papachri@ece.auth.gr](mailto:papachri@ece.auth.gr) |
| Γιώργος Μιχαηλίδης | 40 | geormich@auth.gr |

# Πίνακας Περιεχομένων

Πίνακας Περιεχομένων 3

Λίστα Σχημάτων 4

1. Προτεινόμενη Αρχιτεκτονική Λογισμικού 5

1.1 Αποδόμηση Συστήματος 5

1.2 Απεικόνιση Υλικού/Λογισμικού 5

1.3 Έλεγχος Πρόσβασης και Ασφάλεια 5

2. Πίνακας ιχνηλασιμότητας εγγράφων Σχεδίασης και Απαιτήσεων Λογισμικού 6

3. RESTful API 7

3.1 Endpoints 7

4. SWOT ANALYSIS 10

5. Χρονοδιάγραμμα και Κοστολόγηση Έργου 11

6. Παράρτημα I – Ανοιχτά Θέματα 13

# Λίστα Σχημάτων

[Σχήμα 1. Λεζάντα σχήματα… 5](#_Toc231371493)

# ­­­­Προτεινόμενη Αρχιτεκτονική Λογισμικού

*Το τρίτο κεφάλαιο περιγράφει την αρχιτεκτονική του νέου συστήματος.* ***Ως εισαγωγική ενότητα δείτε την ενότητα 2.3 του εργασίας-παράδειγμα.***

## Αποδόμηση Συστήματος

*Η αποδόμηση του συστήματος περιγράφει την αποδόμηση σε υποσυστήματα και τις ευθύνες του κάθε υποσυστήματος ξεχωριστά. Είναι το βασικό παράγωγο της σχεδίασης του συστήματος.*

***Σημείωση:*** *UML Διαγράμματα Τμημάτων (component diagrams) εδώ!!!*

## Απεικόνιση Υλικού/Λογισμικού

*Η απεικόνιση υλικού/λογισμικού περιγράφει πως τα υποσυστήματα αναφέρονται στο υλικό ή σε άλλα έτοιμα προγράμματα λογισμικού. Ακόμα παραθέτει θέματα που εισάγει η χρησιμοποίηση πολλών κόμβων σε ένα σύστημα ή η επαναχρησιμοποίηση του λογισμικού.*

***Σημείωση:*** *UML Διαγράμματα Ανάπτυξης (deployment diagrams) εδώ!!!*

## Έλεγχος Πρόσβασης και Ασφάλεια

*Ο έλεγχος πρόσβασης και η ασφάλεια περιγράφουν τη μοντελοποίηση των χρηστών του συστήματος με τη μορφή ενός πίνακα πρόσβασης (access matrix). Η παράγραφος αυτή επίσης περιγράφει και διάφορα θέματα ασφαλείας, όπως επιλογή μηχανισμού αυθεντικοποίησης, χρήση κρυπτογράφησης και διαχείριση κλειδιών.*

# Πίνακας ιχνηλασιμότητας εγγράφων Σχεδίασης και Απαιτήσεων Λογισμικού

*Ένας πίνακας ιχνηλασιμότητας με όλες τις αλλαγές που έχουν γίνει από τη μετάβαση στο έγγραφο SDD από το έγγραφο SRD.*

# RESTful API

Τεκμηριώστε εδώ τα διάφορα RESTful APIs. Ένα καλό παράδειγμα τεκμηρίωσης θα βρείτε εδώ: <https://idratherbewriting.com/learnapidoc/docapis_finished_doc_result.html>

Βάλτε υποενότητες για κάθε ξεχωριστό API που έχετε. Αν έχετε 1 web API server το κείμενο θα έχει μία ενότητα. Αν κάνετε microservices αρχιτεκτονική το κείμενο θα έχει όσες ενότητες είναι τα microservice σας. Αν έχετε ένα distributed σύστημα το κείμενο θα έχει τόσες ενότητες όσες είναι τα συστήματά σας.

Π.χ.

## Endpoints

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Method** | | **URL** | | | |
| GET | | surfreport/{beachId} | | | |
| **Description** | | | | | |
| Gets the surf conditions for a specific beach ID. | | | | | |
| **Parameters** | | | | | |
| **Path Parameters** | | | | | |
| Path Parameter | | | Description | | |
| {beachId} | | | Refers to the ID for the beach you want to look up. All Beach ID codes are available from our site at sampleurl.com. | | |
| **Query string parameters** | | | | | |
| Query Parameters | | | Required/Optional | Description | Type |
| Days | | | Optional | The number of days to include in the response. Default is 3. | Integer |
| Time | | | Optional | If you include the time, then only the current hour will be returned in the response. | Integer. Unix format (ms since 1970) in UTC. |
| **Sample Request Body (if exists)** | | | | | |
| Not needed | | | | | |
| **Sample Responses** | | | | | |
| Status | Response | | | | |
| 200 | {  "surfreport": [  {  "beach": "Santa Cruz",  "monday": {  "1pm": {  "tide": 5,  "wind": 15,  "watertemp": 80,  "surfheight": 5,  "recommendation": "Go surfing!"  },  "2pm": {  "tide": -1,  "wind": 1,  "watertemp": 50,  "surfheight": 3,  "recommendation": "Surfing conditions are okay, not great."  },  "3pm": {  "tide": -1,  "wind": 10,  "watertemp": 65,  "surfheight": 1,  "recommendation": "Not a good day for surfing."  }  ...  }  }  ]  } | | | | |
| 400 | {"error":"Please specify database version."} | | | | |
| 404 | {"error":"Beach ID not found"} | | | | |
| 500 | {"error":"Something went wrong. Please try again later."} | | | | |
| XXX | Άλλα σφάλματα που μπορεί αν συμβούν…δείτε εδώ: <https://www.restapitutorial.com/httpstatuscodes.html> | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Method** | | **URL** | | | |
| GET | | <URL> | | | |
| **Description** | | | | | |
| <Description> | | | | | |
| **Parameters** | | | | | |
| **Path Parameters** | | | | | |
| Path Parameter | | | Descriptoin | | |
| <PParam> | | | <Description> | | |
| **Query string parameters** | | | | | |
| Query Parameters | | | Required/Optional | Description | Type |
| <QParam> | | | <Required/Optional> | <Description> | <Type> |
| Sample Request Body | | | | | |
| <Sample body in PUT and POST requests> | | | | | |
| Sample Responses | | | | | |
| Status | Response | | | | |
| <Code> | <Sample JSON> | | | | |
| XXX | Άλλα σφάλματα που μπορεί αν συμβούν…δείτε εδώ: <https://www.restapitutorial.com/httpstatuscodes.html> | | | | |

# Ανάλυση SWOT

**Strengths:**

* Φιλικό για το χρήστη
* Απουσία ανταγωνιστικού προϊόντος
* Χρήση ποικιλίας σχεδιαστικών προγραμμάτων μέσα σε μια πλατφόρμα
* Διευκόλυνση στην επικοινωνία μεταξύ μιας ομάδας σχεδιαστών
* Χρήση απαιτητικών σχεδιαστικών προγραμμάτων μέσω browser

**Weaknesses:**

Μεγάλο κόστος για χρήση σχεδιαστικών προγραμμάτων

Έλλειψη στρατηγικών συνεργασιών με εταιρείες-κατόχους σχεδιαστικών προγραμμάτων

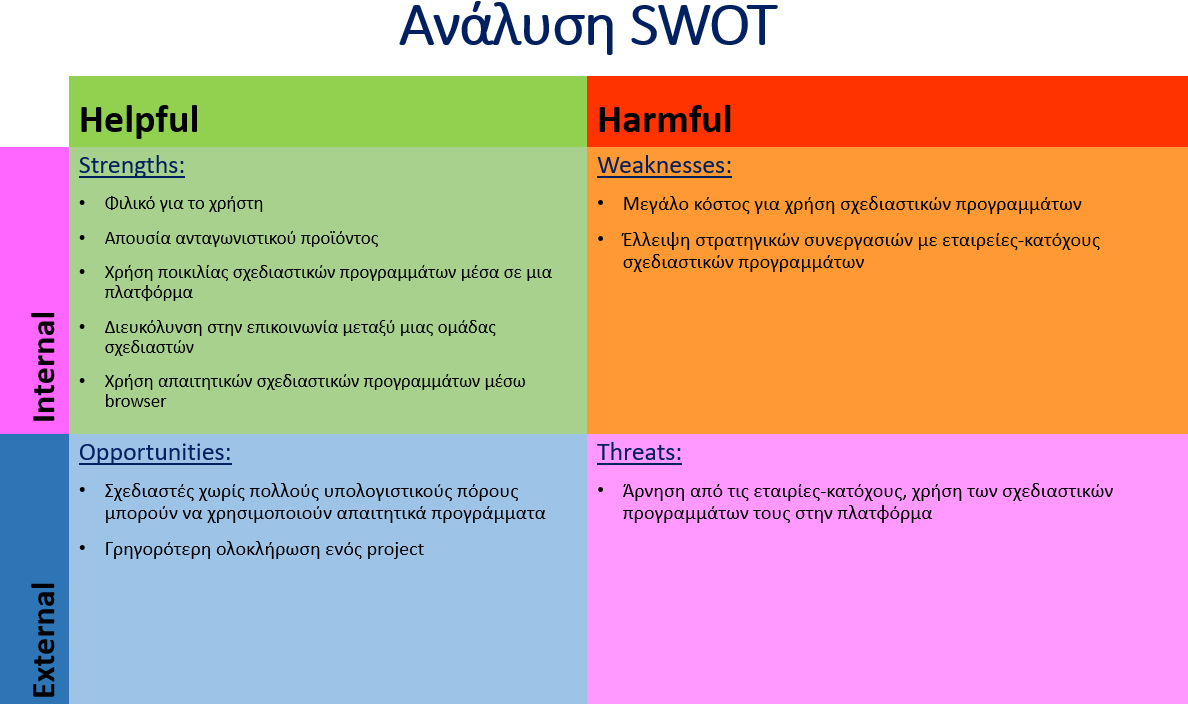
**Opportunities:**

Σχεδιαστές χωρίς πολλούς υπολογιστικούς πόρους μπορούν να χρησιμοποιούν απαιτητικά προγράμματα

Γρηγορότερη ολοκλήρωση ενός project

**Threats:**

Άρνηση από τις εταιρίες-κατόχους, χρήση των σχεδιαστικών προγραμμάτων τους στην πλατφόρμα



# Χρονοδιάγραμμα και Κοστολόγηση Έργου

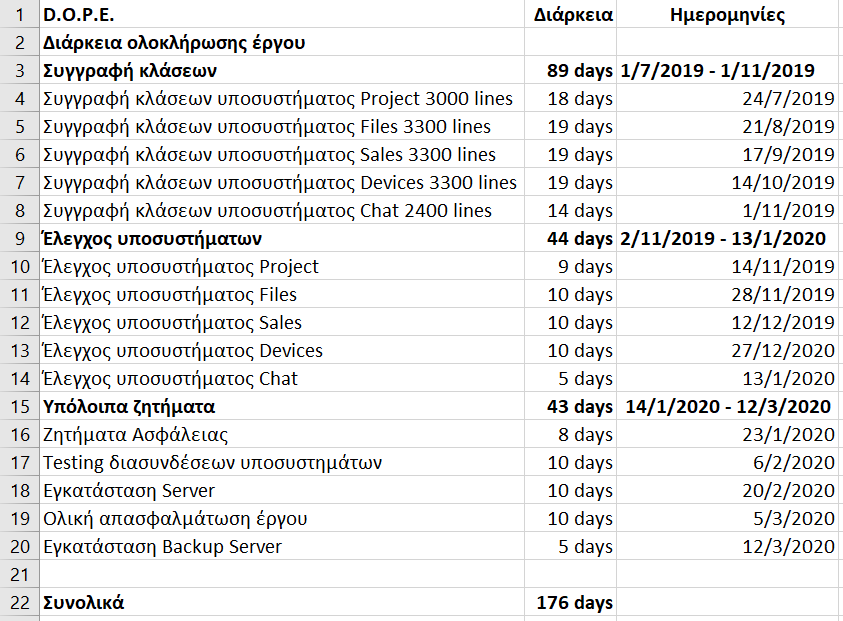
**Χρονοδιάγραμμα:**

Κατά μέσο όρο, ο κάθε μηχανικός της εταιρίας γράφει 1000 πλήρως δοκιμασμένες και αποσφαλματωμένες γραμμές κώδικα το μήνα. Για τις κλάσεις του συστήματος ισχύει:

* 13 Interfaces \* 300 γραμμές κώδικα/Interface = 3900 γραμμές κώδικα
* 5 Entities \* 600 γραμμές κώδικα/Entity = 3000 γραμμές κώδικα
* 9 Controllers \* 600 γραμμές κώδικα/Controller = 5600 γραμμές κώδικα
* 5 Proxies \* 600 γραμμές κώδικα/Proxy = 3000 γραμμές κώδικα

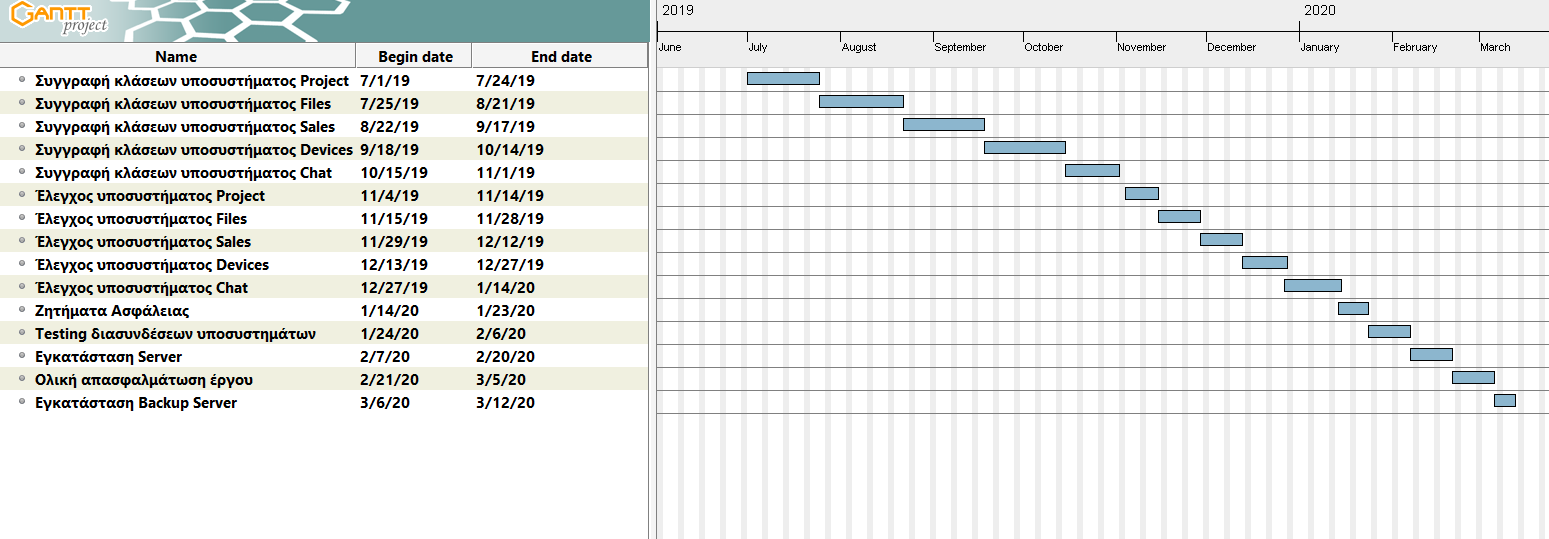
Συνολικά 32 κλάσεις και περίπου 15500 γραμμές κώδικα. Η ομάδα αποτελείται από 4 προγραμματιστές, άρα μπορούν να γράφονται 4000 απoσφαλματωμένες γραμμές κώδικα το μήνα. Άρα 15500/4000 ~= 4 ανθρωπομήνες για την συγγραφή κώδικα. Για τον έλεγχο του συστήματος θα χρειαστούν άλλοι 2 ανθρωπομήνες. Οπότε συνολικά θα χρειαστούν 6 ανθρωπομήνες για την διεκπεραίωση του έργου. Λαμβάνοντας υπόψιν σαββατοκύριακα και αργίες ο τελικός χρόνος είναι 8,5 μήνες.

\*\*Στο χρονοδιάγραμμα συμπεριλαμβάνονται και οι αργίες της 15ης Αυγούστου και των Χριστουγέννων.



**Διάγραμμα Gantt:**

Λόγω του γεγονότος ότι θα δουλεύουμε μόνο εμείς οι 4, η συγγραφή κάθε κλάσης θα γίνεται ταυτόχρονα από όλους μας. Για αυτό και στο διάγραμμα Gantt φαίνεται πως τελειώνει ένα task για να ξεκινήσει ένα άλλο.

****

**Κοστολόγηση έργου:**

Η κοστολόγηση έγινε με βάση την παραδοχή ότι, ο κάθε μηχανικός της εταιρίας γράφει 1000 πλήρως δοκιμασμένες και απασφαλματωμένες γραμμές κώδικα το μήνα.

Για την ολοκλήρωση (integration) ενός συστήματος απαιτούνται 0,3 \* Ν ανθρωπομήνες, όπου 0,3 μια εμπειρική σταθερά και Ν ο αριθμός των κλάσεων που υπάρχουν στο σύστημα.

Οι κλάσεις UI (Interfaces) αποτελούνται κατά μέσο όρο από 300 γραμμές κώδικα η κάθε μία, ενώ οι λειτουργικές κλάσεις (Entities, Controllers, Proxies) από 600 γραμμές κώδικα η κάθε μία.

Άρα το συνολικό κόστος προκύπτει από τους ανθρωπομήνες που χρειάζονται για την συγγραφή κώδικα, τον έλεγχο των υποσυστημάτων, την διευθέτηση των ζητημάτων ασφάλειας, το test των διασυνδεδεμένων υποσυστημάτων, την εγκατάσταση του server και του backup server και την ολική αποσφαλμάτωση του συστήματος. Για την ολοκλήρωση του project χρειάζεται να καλυφθούν κάποια επιπλέον κόστη όπως αυτά της κτηριακής εγκατάστασης και του στησίματος και συντήρησης του server.

**Οπότε:**

* **Ανθρώπινο δυναμικό:** 6 ανθρωπομήνες \* 4 άτομα \* 2000 €/μήνα = 48000 €
* **Κτηριακές εγκαταστάσεις:** 5000€
* **Στήσιμο και συντήρηση Servers:**
* Server: 2000€/έτος
* Backup server: 1000€/έτος

**Συνολικά 55000 €**

# Παράρτημα I – Ανοιχτά Θέματα

*Λίστα με ανοιχτά θέματα που θα συζητηθούν με τον πελάτη πριν την οριστικοποίηση (πρώτη σταθερή έκδοση) του Εγγράφου Σχεδίασης Συστήματος.*