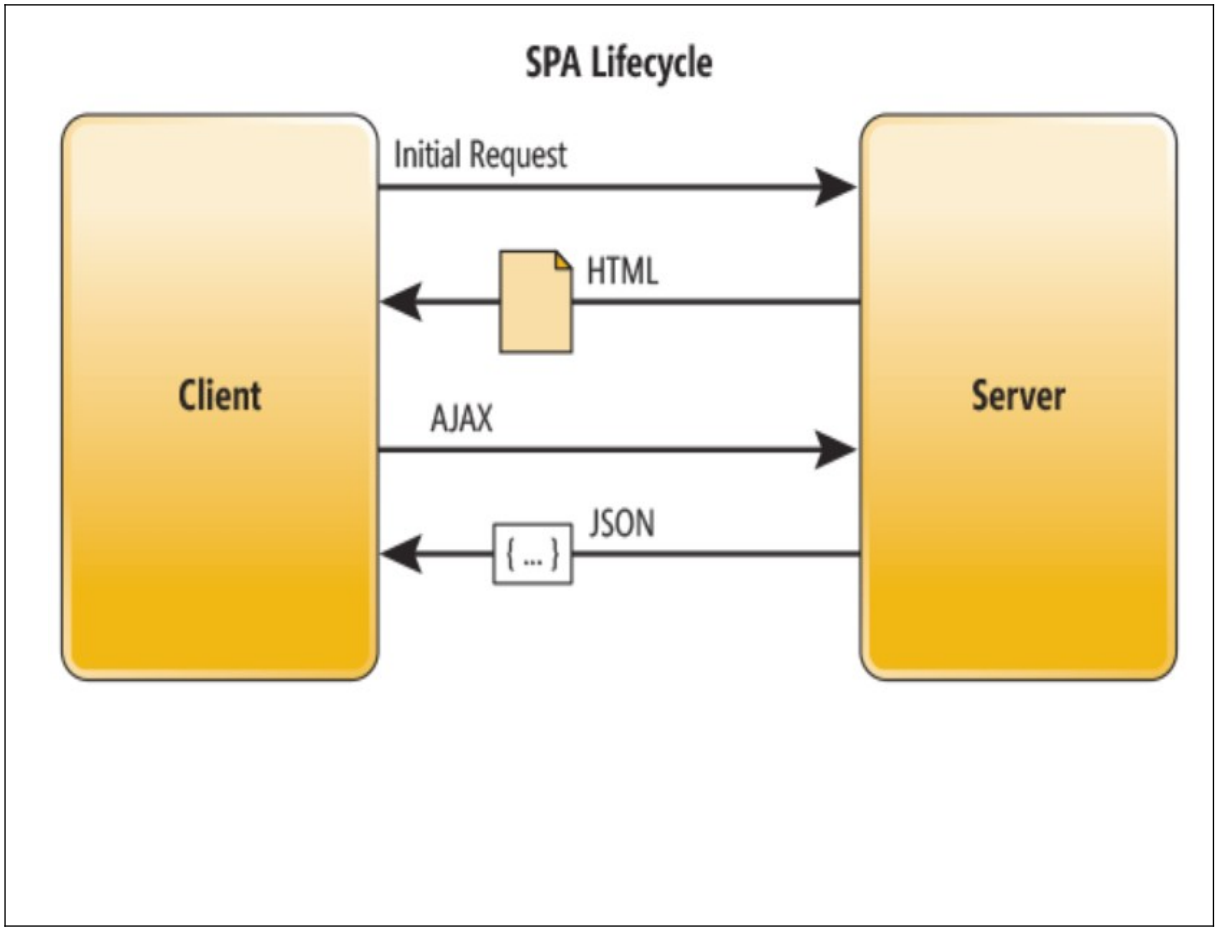
ΔΙΕΘΝΕΣ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΤΗΣ ΕΛΛΑΔΟΣ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

«Ανάπτυξη Mobile εφαρμογών ως SPA (Single Page Applications)»



Του φοιτητή
Μπατζιάκα Κωνσταντίνου
Αρ. Μητρώου: 134036

Επιβλέπων
Δημήτριος Τεκτονίδης
Διδάσκων

Ημερομηνία 10/9/2021

Ανάπτυξη Mobile εφαρμογών ως SPA (Single Page Applications)
#4413
Μπατζιάκας Κωνσταντίνος
Τεκτονίδης Δημήτριος
29-03-2021
Ημερομηνία περάτωσης Δ.Ε. 10/9/2021

Βεβαιώνω ότι είμαι ο συγγραφέας αυτής της εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης, έχω καταγράψει τις όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών, εικόνων και κειμένου, είτε αυτές αναφέρονται ακριβώς είτε παραφρασμένες. Επιπλέον, βεβαιώνω ότι αυτή η εργασία προετοιμάστηκε από εμένα προσωπικά, ειδικά ως διπλωματική εργασία, στο Τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του ΔΙ.ΠΑ.Ε.

«Στην Εύα»

Πρόλογος

Τα Single Page Applications (SPA) είναι web εφαρμογές όπου αλληλεπιδρούν με τον χρήστη ξαναγράφοντας το περιεχόμενο της τρέχουσας ιστοσελίδας με νέα δεδομένα απο τον web server, αντιθέτως με την κλασσική μέθοδο των web εφαρμογών οπού φορτώνουν ολόκληρες νέες σελίδες. Ο Στόχος είναι η γρήγορη μετάβαση, όπου κάνουν μία ιστοσελίδα να μοιάζει περισσότερο με native εφαρμογή. Στα SPA δεν λαμβάνει χώρα η ανανέωση της ιστοσελίδας, αντιθέτως όλος ο απαραίτητος HTML, CSS, και JavaScript κώδικας ανακτάται από τον browser με την φόρτωση της σελίδας[1], είτε οι κατάλληλοι πόροι φορτώνονται δυναμικά και προσθέτονται στην ιστοσελίδα όπως απαιτείται, συνήθως ως ανταπόκριση στις ενέργειες του χρήστη. Η σελίδα δεν φορτώνεται ξανά σε κανένα σημείο της διαδικασίας, ούτε μεταφέρει τον έλεγχο σε άλλη σελίδα, αν και ο κατακερματισμός τοποθεσίας ή το API μπορούν να χρησιμοποιηθούν για να παρέχουν τη δυνατότητα πλοήγησης ξεχωριστών λογικών σελίδων στην εφαρμογή[2]. Τα Progressive Web Applications (PWA) είναι ένας τύπος λογισμικού εφαρμογών που παραδίδεται μέσω του διαδικτύου, που έχει δημιουργηθεί με χρήση κοινών τεχνολογιών ιστού, όπως HTML, CSS και JavaScript. Προορίζεται να λειτουργήσει σε οποιαδήποτε πλατφόρμα που χρησιμοποιεί πρόγραμμα περιήγησης που συμμορφώνεται με τα πρότυπα, συμπεριλαμβανομένων επιτραπέζιων και κινητών συσκευών. Δεδομένου ότι ενα Progressive Web Application είναι ένας τύπος ιστοσελίδας ή ιστότοπος γνωστός ως εφαρμογή ιστού, δεν απαιτεί ξεχωριστή ομαδοποίηση ή διανομή. Οι προγραμματιστές μπορούν απλά να δημοσιεύσουν την διαδικτυακή εφαρμογή στο διαδίκτυο, να διασφαλίσουν ότι πληροί τις βασικές "απαιτήσεις εγκατάστασης" και οι χρήστες θα μπορούν να προσθέσουν την εφαρμογή στην αρχική τους οθόνη.

Περίληψη

Στόχος της πτυχιακής μου εργασίας είναι να παρουσιάσω τα αποτελέσματα της εφαρμογής των SPA με την χρήση του JavaScript Framework React.js και των .NET Core και Entity Framework καθώς και της χρησιμότητας αυτών, μέσω μίας native εφαρμογής. Η εφαρμογή Thessaloniki Club περιέχει προσφορές απο καταστήματα της Θεσσαλονίκης, όπου ο χρήστης μπορεί να κάνει κράτηση μέσω της εφαρμογής και να αγοράσει την προσφορά. Στην εφαρμογή χρησιμοποιούνται βιβλιοθήκες του React.js, καθώς και το ECMAScript 6, γνωστό και ως ECMAScript 2015 όπου είναι η τελευταία έκδοση του ECMAScript προτύπου και χρησιμοποείται απο την JavaScript.

«Mobile application development as Single Page Application (SPA)»

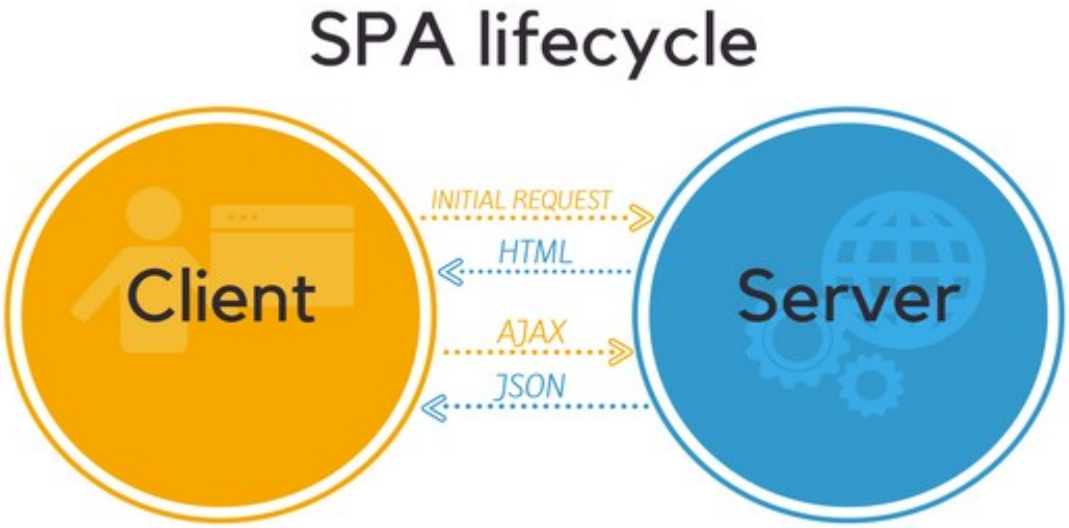
«Konstantinos Batziakas»

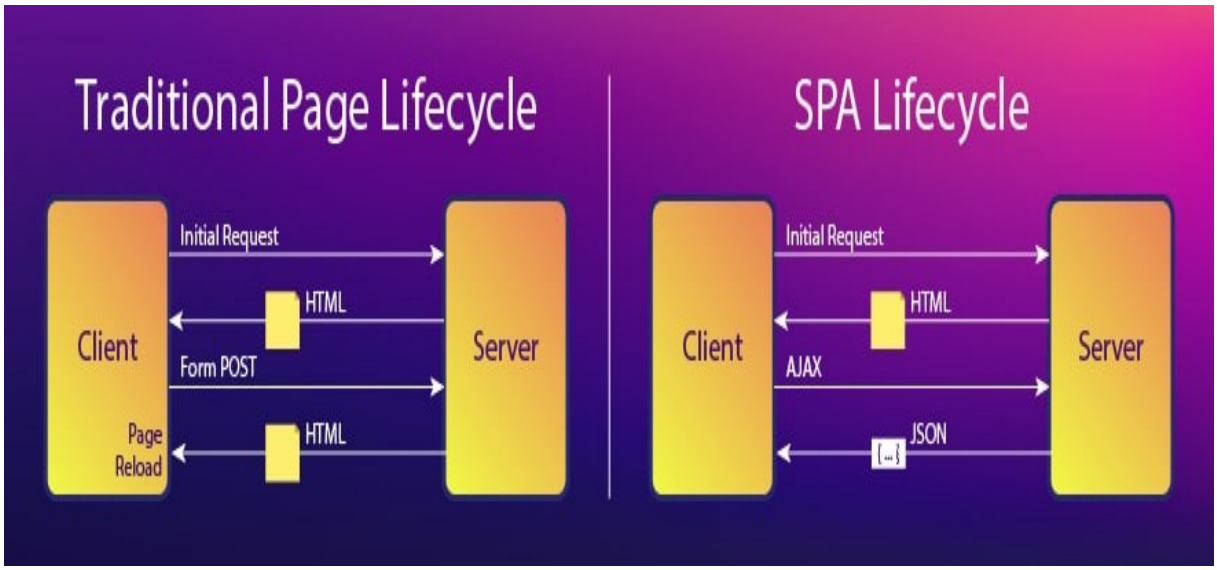
Ευχαριστίες

Ένα μεγάλο ευχαριστώ στους ανθρώπους που ήταν δίπλα μου, την οικογένεια μου, την Εύα, τον Νίκο.

Περιεχόμενα

Κατάλογος Σχημάτων




Traditional Page Lifecycle
Client
Server
Initial Request
HTML
Form POST
HTML
Page Reload
SPA Lifecycle
Client
Server
Initial Request
HTML
AJAX
JSON

Συντομογραφίες

Δ.Ε.	Διπλωματική Εργασία
ΔΙΠΑΕ	Διεθνές Πανεπιστήμιο Ελλάδος
Π.Ε.	Πτυχιακή Εργασία
SPA	Single Page Application
PWA	Progressive Web Application
DOM	Document Object Model
AJAX	Asynchronous JavaScript And XML
JSON	JavaScript Object Notation

1 Single Page Applications (SPA)

1.1 Εισαγωγή

Στο πρώτο κεφάλαιο θα μελετηθούν τα Single Page Applications (SPA).

1.2 Ιστορία

Η προέλευση του όρου εφαρμογή μιας σελίδας είναι ασαφής, αν και η έννοια συζητήθηκε τουλάχιστον το 2003[3]. Ο Στιούαρτ Μόρις, φοιτητής προγραμματισμού στο Πανεπιστήμιο του Κάρντιφ της Ουαλίας, έγραψε τον αυτοδύναμο ιστότοπο στο slashdotslash.com με τους ίδιους στόχους και λειτουργίες τον Απρίλιο του 2002[4], και αργότερα τον ίδιο χρόνο οι Λούκας Μπερντό, Κέβιν Χάκμαν, Μάικλ Πίτσι και Κλίφορντ Yeh περιέγραψαν μια SPA εφαρμογή στην ευρεσιτεχνία ΗΠΑ 8.136.109[5].

1.3 Τεχνικές προσεγγίσεις

Υπάρχουν διάφορες διαθέσιμες τεχνικές που επιτρέπουν στο πρόγραμμα περιήγησης να διατηρήσει μία SPA ακόμη και όταν η εφαρμογή απαιτεί επικοινωνία διακομιστή.

1.4 JavaScript frameworks

Τα JavaScript Web Frameworks καθώς και οι JavaScript Libraries, όπως AngularJS, Ember.js, ExtJS, Knockout.js, Meteor.js, React,Vue.js και Svelte έχουν υιοθετήσει τις αρχές των SPA. Εκτός του ExtJS, όλα τα υπόλοιπα ειναι ανοιχτού κώδικα.

- AngularJS: Είναι ένα πλήρες client-side framework. Το πρότυπο της AngularJS βασίζεται στη σύνδεση δεδομένων αμφίδρομης Data-Binding. Το Data-Binding είναι ένας αυτόματος τρόπος ενημέρωσης της προβολής δεδομένων κάθε φορά που αλλάζει το μοντέλο, καθώς και ενημέρωση του μοντέλου κάθε φορά που αλλάζει η προβολή. Το πρότυπο HTML μεταγλωτίζεται στο πρόγραμμα περιήγησης. Το βήμα της μεταγλώττισης δημιουργεί καθαρό HTML, το οποίο το πρόγραμμα περιήγησης το μετατρέπει εκ νέου σε ζωντανή προβολή. Το βήμα επαναλαμβάνεται για τις επόμενες προβολές σελίδας. Στο παραδοσιακό server-side HTML από την πλευρά του διακομιστή, έννοιες όπως ο controller και το model αλληλεπιδρούν μέσα σε μια διαδικασία διακομιστή για να παράγουν νέες προβολές HTML. Στο AngularJS, οι καταστάσεις controller και model διατηρούνται εντός του προγράμματος περιήγησης. Επομένως, νέες σελίδες μπορούν να δημιουργηθούν χωρίς καμία αλληλεπίδραση με διακομιστή.
- Ember.js: είναι JavaScript Framework από την πλευρά του πελάτη που βασίζεται στο αρχιτεκτονικό μοτίβο λογισμικού model-view-controller (MVC). Επιτρέπει στους προγραμματιστές να δημιουργούν scallable single page εφαρμογές ενσωματώνοντας κοινά ιδιώματα και βέλτιστες πρακτικές σε ένα framework που παρέχει ένα πλούσιο object model, δηλωτική αμφίδρομη δέσμευση δεδομένων, υπολογισμένες ιδιότητες, αυτόματη ενημέρωση των templates που υποστηρίζονται από το Handlebars.js και ένα δρομολογητή για διαχείριση του application state.
- ExtJS: είναι επίσης ένα client-side framework που επιτρέπει τη δημιουργία εφαρμογών MVC. Διαθέτει το δικό του event system και διάφορα στοιχεία UI (grids, παράθυρα διαλόγου, φόρμες κ.λπ.). Διαθέτει το δικό του σύστημα τάξης με δυναμικό ή στατικό φορτωτή. Η

εφαρμογή που έχει δημιουργηθεί με ExtJS μπορεί είτε να υπάρχει μόνη της (με κατάσταση στο πρόγραμμα περιήγησης) είτε με το διακομιστή (π.χ. με το REST API που χρησιμοποιείται για να γεμίσει τα εσωτερικά του καταστήματα). Το ExtJS έχει ενσωματωμένες μόνο δυνατότητες χρήσης τοπικού αποθηκευτικού χώρου, επομένως οι μεγαλύτερες εφαρμογές χρειάζονται διακομιστή για αποθήκευση της κατάστασης.

- Knockout.js είναι ένα client-side JavaScript framework όπου χρησιμοποιεί templates βασισμένα στο πρότυπο model-view-ViewModel (MVVM).
- Meteor.js είναι ενα full-stack (client-server) JavaScript framework σχεδιασμένο αποκλειστικά για SPAs. Διαθέτει απλούστερη δέσμευση δεδομένων από το Angular, Ember ή ReactJS[6], και χρησιμοποιεί το πρωτόκολλο κατανεμημένων δεδομένων[7] και ένα μοτίβο δημοσίευσης-εγγραφής για να διαδίδει αυτόματα αλλαγές δεδομένων στους clients σε πραγματικό χρόνο χωρίς να απαιτείται από τον προγραμματιστή να γράψει οποιονδήποτε κωδικό συγχρονισμού. Το full-stack διασφαλίζει ότι όλα τα επίπεδα, από τη βάση δεδομένων έως τα πρότυπα, ενημερώνονται αυτόματα όταν είναι απαραίτητο. Τα πακέτα οικοσυστήματος, όπως το Server Side Rendering[8], αντιμετωπίζουν το πρόβλημα της βελτιστοποίησης μηχανών αναζήτησης.
- React.js είναι μία βιβλιοθήκη της JavaScript όπου χρημοποιείται για την υλοποίηση διεπαφών χρήστη. Αναπτύχθηκε και συντηρείται απο το Facebook. Το React χρησιμοποιεί μια νέα γλώσσα που είναι ένα μείγμα JavaScript και HTML (υποσύνολο HTML). Αρκετές εταιρείες χρησιμοποιούν το React μαζί με την Redux (βιβλιοθήκη JavaScript), που προσθέτει δυνατότητες διαχείρισης κατάστασης, το οποίο (με αρκετές άλλες βιβλιοθήκες) επιτρέπει στους προγραμματιστές να δημιουργούν πολύπλοκες εφαρμογές[9].
- Vue.js είναι επίσης ένα JavaScript framework για υλοποίηση διεπαφών χρήστη. Όπως και στην react.js, οι vue developers χρησμοποιούν την Vuex (JavaScript βιβλιοθήκη) για την διαχείρηση κατάστασης.
- Το Svelte είναι ένα framework για τη δημιουργία διεπαφών χρήστη που συγκεντρώνει κώδικα Svelte σε χειρισμούς DOM JavaScript, αποφεύγοντας την ανάγκη ομαδοποίησης ενός framework στον πελάτη και επιτρέποντας απλούστερη σύνταξη ανάπτυξης εφαρμογών.

1.5 Ajax

Από το 2006, η πιο εξέχουσα τεχνική που χρησιμοποιήθηκε ήταν το Ajax. Το Ajax περιλαμβάνει τη χρήση ασύγχρονων αιτημάτων στον διακομιστή για δεδομένα XML ή JSON, όπως με το XMLHttpRequest της JavaScript ή πιο σύγχρονα την fetch() (από το 2017) ή το καταργημένο αντικείμενο ActiveX. Σε αντίθεση με τη δηλωτική προσέγγιση των περισσότερων πλαισίων SPA, με τον Ajax ο ιστότοπος χρησιμοποιεί απευθείας JavaScript ή μια βιβλιοθήκη JavaScript όπως το jQuery για να χειριστεί το DOM και να επεξεργαστεί στοιχεία HTML. Το Ajax έχει διαδοθεί περαιτέρω από βιβλιοθήκες όπως το jQuery, το οποίο παρέχει μια απλούστερη σύνταξη και ομαλοποιεί τη συμπεριφορά του Ajax σε διαφορετικά προγράμματα περιήγησης που ιστορικά είχαν διαφορετική συμπεριφορά.

1.6 Μεταφορά δεδομένων

Τα αιτήματα προς τον διακομιστή συνήθως έχουν ως αποτέλεσμα είτε ανεπεξέργαστα δεδομένα (π.χ. XML ή JSON) είτε επιστροφή νέου HTML. Στην περίπτωση που το HTML επιστρέφεται από τον διακομιστή, το JavaScript στο πρόγραμμα - πελάτη ενημερώνει μια μερική περιοχή του DOM[10]. Όταν επιστρέφονται ακατέργαστα δεδομένα, συχνά χρησιμοποιείται μια διαδικασία JavaScript XML από τον πελάτη και στην περίπτωση JSON ένα πρότυπο για τη μετάφραση των ακατέργαστων δεδομένων σε HTML, η οποία στη συνέχεια χρησιμοποιείται για την ενημέρωση μερικής περιοχής του DOM .

1.7 Αρχιτεκτονική διακομιστή

- Thin server architecture: Τα SPA μετακινούν τα δεδομένα από τον διακομιστή στον πελάτη, με τον ρόλο του διακομιστή να εξελίσσεται σε καθαρό API δεδομένων η web υπηρεσία. Αυτή η αρχιτεκτονική, σε ορισμένους κύκλους, αποκαλείται "Thin Server Architecture" για να τονίσει ότι η πολυπλοκότητα έχει μεταφερθεί από τον διακομιστή στον πελάτη, με το επιχείρημα ότι αυτό τελικά μειώνει τη συνολική πολυπλοκότητα του συστήματος.
- Thick stateful server architecture: Ο διακομιστής διατηρεί την απαραίτητη κατάσταση στη μνήμη της κατάστασης προγράμματος - πελάτη της σελίδας. Με αυτόν τον τρόπο, όταν οποιοδήποτε αίτημα χτυπήσει τον διακομιστή (συνήθως ενέργειες χρήστη), ο διακομιστής στέλνει την κατάλληλη HTML ή/και JavaScript με συγκεκριμένες αλλαγές για να φέρει τον πελάτη στη νέα επιθυμητή κατάσταση (συνήθως προσθέτοντας/διαγράφοντας/ενημερώνοντας ένα μέρος του πελάτη DOM). Ταυτόχρονα, ενημερώνεται η κατάσταση στο διακομιστή. Το μεγαλύτερο μέρος της λογικής εκτελείται στον διακομιστή και η HTML συνήθως αποδίδεται επίσης στον διακομιστή. Κατά κάποιο τρόπο, ο διακομιστής προσομοιώνει ένα πρόγραμμα περιήγησης ιστού, λαμβάνει συμβάντα και πραγματοποιεί αλλαγές στην κατάσταση διακομιστή, οι οποίες διαδίδονται αυτόματα στον πελάτη. Αυτή η προσέγγιση χρειάζεται περισσότερη μνήμη διακομιστή και επεξεργασία διακομιστή, αλλά το πλεονέκτημα είναι ότι είναι ένα απλοποιημένο μοντέλο ανάπτυξης επειδή α) η εφαρμογή είναι συνήθως πλήρως κωδικοποιημένη στον διακομιστή και β) τα δεδομένα και η κατάσταση διεπαφής χρήστη στο διακομιστή μοιράζονται στον ίδιο χώρο μνήμης χωρίς ανάγκη για προσαρμοσμένες γέφυρες επικοινωνίας πελάτη/διακομιστή
- Thick stateless server architecture: Η σελίδα πελάτη στέλνει δεδομένα που αντιπροσωπεύουν την τρέχουσα κατάσταση στον διακομιστή, συνήθως μέσω αιτημάτων Ajax. Χρησιμοποιώντας αυτά τα δεδομένα, ο διακομιστής είναι σε θέση να ανασυνθέσει την κατάσταση του πελάτη του τμήματος της σελίδας που πρέπει να τροποποιηθεί και μπορεί να δημιουργήσει τα απαραίτητα δεδομένα ή κώδικα (για παράδειγμα, ως JSON ή JavaScript), τα οποία επιστρέφονται στον πελάτη για να τον φέρει σε μια νέα κατάσταση, συνήθως τροποποιώντας την δομή της σελίδας DOM σύμφωνα με τη δράση του πελάτη που παρακίνησε το αίτημα. Αυτή η προσέγγιση απαιτεί την αποστολή περισσότερων δεδομένων στον διακομιστή και μπορεί να απαιτεί περισσότερους υπολογιστικούς πόρους ανά αίτημα για την μερική ή πλήρη ανακατασκευή της κατάστασης της σελίδας πελάτη στον διακομιστή. Ταυτόχρονα, αυτή η προσέγγιση είναι πιο εύκολα επεκτάσιμη επειδή δεν υπάρχουν δεδομένα σελίδας ανά πελάτη στον διακομιστή και, ως εκ τούτου, τα αιτήματα του Ajax μπορούν να αποσταλούν σε διαφορετικούς κόμβους διακομιστή χωρίς να απαιτείται κοινή χρήση δεδομένων συνεδρίας.

1.8 Τοπική εκτέλεση

Ορισμένα SPA μπορούν να εκτελεστούν από ένα τοπικό αρχείο χρησιμοποιώντας το σχήμα URI. Αυτό δίνει στους χρήστες τη δυνατότητα να κατεβάσουν το SPA από έναν διακομιστή και να εκτελέσουν το αρχείο από μια τοπική συσκευή αποθήκευσης, χωρίς να εξαρτώνται από τη συνδεσιμότητα διακομιστή. Εάν ένα τέτοιο SPA θέλει να αποθηκεύει και να ενημερώνει δεδομένα, πρέπει να χρησιμοποιεί Web Storage που βασίζεται σε πρόγραμμα περιήγησης. Αυτές οι εφαρμογές επωφελούνται από τις προόδους που διατίθενται με το HTML5.

1.9 Κύκλος ζωής σελίδας

Ένα SPA φορτώνεται πλήρως στην αρχική φόρτωση της σελίδας και στη συνέχεια οι περιοχές σελίδας αντικαθίστανται ή ενημερώνονται με νέα τμήματα σελίδων που φορτώνονται από τον διακομιστή κατόπιν αιτήματος. Για να αποφευχθεί η υπερβολική λήψη αχρησιμοποίητων δυνατοτήτων, ένα SPA θα κατεβάζει συχνά και προοδευτικά περισσότερες λειτουργίες όταν

απαιτούνται, είτε μικρά κομμάτια της σελίδας είτε πλήρεις ενότητες οθόνης. Με αυτόν τον τρόπο υπάρχει μια αναλογία μεταξύ "καταστάσεων" σε ένα SPA και "σελίδων" σε έναν παραδοσιακό ιστότοπο. Επειδή η "κατάσταση πλοήγησης" στην ίδια σελίδα είναι ανάλογη με την πλοήγηση σελίδων, θεωρητικά, κάθε ιστότοπος που βασίζεται σε σελίδα θα μπορούσε να μετατραπεί σε μία σελίδα αντικαθιστώντας στην ίδια σελίδα μόνο τα αλλαγμένα μέρη.

2 Progressive Web Applications (PWA)

2.1 Εισαγωγή

Στο δεύτερο κεφάλαιο θα εισαχθεί η έννοια των Progressive Web Applications (PWA).

2.2 Ιστορία

2.2.1 Προκάτοχοι

Κατά την κυκλοφορία του iPhone το 2007, ο Steve Jobs ανακοίνωσε ότι οι εφαρμογές ιστού (που αναπτύχθηκαν σε HTML5 χρησιμοποιώντας αρχιτεκτονική AJAX) θα ήταν η τυπική μορφή για τις εφαρμογές iPhone. Δεν απαιτήθηκε κιτ ανάπτυξης λογισμικού (SDK) και οι εφαρμογές θα ενσωματώνονταν πλήρως στη συσκευή μέσω της μηχανής περιήγησης Safari. [11] Αυτό το μοντέλο άλλαξε αργότερα για το App Store, ως μέσο πρόληψης των jailbreakers. [12] Τον Οκτώβριο του 2007 ο Jobs ανακοίνωσε ότι ένα SDK θα ξεκινήσει το επόμενο έτος. [11] Ως αποτέλεσμα, αν και η Apple συνέχισε να υποστηρίζει webapps, η συντριπτική πλειοψηφία των εφαρμογών iOS στράφηκε προς το App Store. Από τις αρχές της δεκαετίας του 2010, οι δυναμικές ιστοσελίδες επέτρεψαν τη χρήση τεχνολογιών ιστού για τη δημιουργία διαδραστικών εφαρμογών ιστού. Το responsive web desgin και η ευελιξία μεγέθους οθόνης που παρέχει, έκαναν την υλοποίηση των PWA πιο προσβάσιμη. Οι συνεχείς βελτιώσεις σε HTML, CSS και JavaScript επέτρεψαν στις διαδικτυακές εφαρμογές να ενσωματώσουν μεγαλύτερα επίπεδα διαδραστικότητας, καθιστώντας δυνατές τις εμπειρίες που μοιάζουν με τη γλώσσα σε έναν ιστότοπο. [13]. Το 2013, η Mozilla κυκλοφόρησε το Firefox OS. Προοριζόταν ως ένα λειτουργικό σύστημα ανοιχτού κώδικα για την εκτέλεση webapps ως εγγενών εφαρμογών σε φορητές συσκευές. Το Firefox OS βασίστηκε στη μηχανή απόδοσης Gecko με μια διεπαφή χρήστη που ονομάζεται Gaia, γραμμένη σε HTML5. Η ανάπτυξη του Firefox OS έληξε το 2016, και το έργο διακόπηκε εντελώς το 2017, [14] αν και ένα τμήμα του Firefox OS χρησιμοποιήθηκε ως βάση του KaiOS, μιας πλατφόρμας τηλεφώνου χαρακτηριστικών. [15]

2.2.2 Αρχική εισαγωγή

Το 2015, ο σχεδιαστής Frances Berriman και ο μηχανικός του Google Chrome Alex Russell επινόησαν τον όρο "Progressive Web Apps" [16] για να περιγράψουν εφαρμογές που εκμεταλλεύονται νέες δυνατότητες που υποστηρίζονται από σύγχρονα προγράμματα περιήγησης, συμπεριλαμβανομένων των service workers και των web app manifests, που επιτρέπουν στους χρήστες να αναβαθμίζουν εφαρμογές ιστού σε progressive web applications στο εγγενές λειτουργικό τους σύστημα (OS). Στη συνέχεια, η Google κατέβαλε σημαντικές προσπάθειες για την προώθηση της ανάπτυξης PWA για Android. [17] [18] Ο Firefox εισήγαγε υποστήριξη για εργαζόμενους στην υπηρεσία το 2016 και ακολούθησαν οι Microsoft Edge και Apple Safari το 2018, [19] [17] καθιστώντας τους service workers διαθέσιμους σε όλα τα μεγάλα συστήματα.

2.3 Χαρακτηριστικά

Τα Progressive Web Applications έχουν σχεδιαστεί για να λειτουργούν σε οποιοδήποτε πρόγραμμα περιήγησης που συμμορφώνεται με τα κατάλληλα πρότυπα ιστού. Όπως και με άλλες λύσεις πολλαπλών πλατφορμών, ο στόχος είναι να βοηθήσουμε τους προγραμματιστές να δημιουργήσουν εφαρμογές μεταξύ πλατφορμών πιο εύκολα από ό, τι με τις εγγενείς εφαρμογές. [17] Ορισμένα Progressive Web Applications χρησιμοποιούν μια αρχιτεκτονική προσέγγιση που ονομάζεται Μοντέλο App Shell. [20] Σε αυτό το μοντέλο, τα service workers αποθηκεύουν τη Βασική διεπαφή χρήστη ή το «κέλυφος» της responsive web application στην προσωρινή μνήμη εκτός σύνδεσης του προγράμματος περιήγησης. Αυτό το μοντέλο επιτρέπει στα PWA να διατηρούν χρήση native φύσης με ή χωρίς συνδεσιμότητα ιστού. Αυτό μπορεί να βελτιώσει το χρόνο φόρτωσης, παρέχοντας ένα αρχικό στατικό πλαίσιο, μια διάταξη ή αρχιτεκτονική στην οποία το περιεχόμενο μπορεί να φορτωθεί προοδευτικά αλλά και δυναμικά. [20]

2.4 Τεχνικά κριτήρια

Τα τεχνικά βασικά κριτήρια για έναν ιστότοπο που πρέπει να θεωρείται Progressive Web Application και ως εκ τούτου "installable" από προγράμματα περιήγησης περιγράφηκαν από τον Russell σε μια συνέχεια ανάρτησης [21]:

- Προέρχονται από ασφαλή προέλευση. Δίνονται πάνω από το TLS πρωτόκολλο και από green padlock displays (χωρίς native περιεχόμενο). Τα Progressive Web Applications πρέπει να προβάλλονται μέσω HTTPS για να διασφαλίζεται το απόρρητο, η ασφάλεια και η γνησιότητα του περιεχομένου των χρηστών.
- Καταχώρηση ενός service worker σε έναν fetch handler. Τα Progressive Web Applications πρέπει να χρησιμοποιούν service workers για να δημιουργήσουν προγραμματιζόμενες κρυφές μνήμες περιεχομένου. Σε αντίθεση με την κανονική κρυφή μνήμη ιστού HTTP, η οποία αποθηκεύει το περιεχόμενο μετά την πρώτη χρήση και στη συνέχεια βασίζεται σε διάφορες ευρετικές πληροφορίες για να μαντέψει πότε το περιεχόμενο δεν είναι πλέον απαραίτητο, οι προγραμματιζόμενες κρυφές μνήμες μπορούν να προαναφέρουν ρητά το περιεχόμενο εκ των προτέρων πριν χρησιμοποιηθεί για πρώτη φορά και να το απορρίψουν ρητά όταν είναι δεν χρειάζεται πλέον. [22] Αυτή η απαίτηση βοηθά τις σελίδες να είναι προσβάσιμες εκτός σύνδεσης ή σε δίκτυα χαμηλής ποιότητας.
- Αναφορά σε ένα web app manifest. Το manifest πρέπει να περιέχει τουλάχιστον τις πέντε βασικές ιδιότητες: όνομα ή short_name, start_url και οθόνη (με τιμή αυτόνομης, πλήρους οθόνης ή ελάχιστο ui) και εικονίδια (με εκδόσεις 192px και 512px). Λόγω της ύπαρξης manifest, τα PWA μπορούν εύκολα να κοινοποιηθούν μέσω μιας διεύθυνσης URL, να ανακαλυφθούν από μια μηχανή αναζήτησης και δεν απαιτούν πολύπλοκη εγκατάσταση (αλλά μπορούν να καταχωριστούν σε κατάστημα εφαρμογών τρίτων). [23] Επιπλέον, τα PWA υποστηρίζουν εγγενείς αλληλεπιδράσεις και πλοήγηση σε στιλ εφαρμογής, συμπεριλαμβανομένης της προσθήκης στην αρχική οθόνη, προβολή οθονών splashs κ.λπ.

2.5 Τεχνολογίες

Υπάρχουν πολλές τεχνολογίες που χρησιμοποιούνται συνήθως για τη δημιουργία Progressive Web Applications. Μια διαδικτυακή εφαρμογή θεωρείται PWA εάν πληροί τα "τεχνικά κριτήρια" και έτσι μπορεί να λειτουργήσει εκτός σύνδεσης και μπορεί να προστεθεί στην αρχική οθόνη της

συσκευής. Για να ικανοποιηθεί αυτός ο ορισμός, όλα τα PWA απαιτούν τουλάχιστον έναν service worker και ένα manifest. [24]

2.6 Manifest

Το web app manifest[25] είναι μια προδιαγραφή W3C που ορίζει ένα manifest που βασίζεται σε JSON (συνήθως επισημαίνεται manifest.json) [23] για να παρέχει στους προγραμματιστές ένα κεντρικό σημείο για την τοποθέτηση μεταδεδομένων που σχετίζονται με μια εφαρμογή ιστού, συμπεριλαμβανομένων:

- Το όνομα της διαδικτυακής εφαρμογής
- Σύνδεσμοι προς τα εικονίδια της εφαρμογής ιστού ή αντικείμενα εικόνας
- Η προτιμώμενη διεύθυνση URL για την εκκίνηση ή το άνοιγμα της διαδικτυακής εφαρμογής
- Τα δεδομένα διαμόρφωσης της εφαρμογής ιστού
- Προεπιλεγμένος προσανατολισμός της εφαρμογής Ιστού
- Η επιλογή ρύθμισης της λειτουργίας εμφάνισης

Αυτά τα μεταδεδομένα είναι ζωτικής σημασίας για μια εφαρμογή να προστεθεί στην αρχική οθόνη ή να καταχωρηθεί με άλλο τρόπο παράλληλα με τις native εφαρμογές.

2.7 Service workers

Ένας service worker είναι ένας web worker που υλοποιεί έναν προγραμματιζόμενο διακομιστή μεσολάβησης που μπορεί να απαντήσει σε αιτήματα ιστού/HTTP του κύριου εγγράφου. Είναι σε θέση να ελέγξει τη διαθεσιμότητα ενός απομακρυσμένου διακομιστή και να αποθηκεύσει προσωρινά το περιεχόμενο όταν είναι διαθέσιμος αυτός ο διακομιστής και να το εμφανίσει αργότερα στο έγγραφο. Οι εργαζόμενοι σε υπηρεσίες, όπως και κάθε άλλος εργαζόμενος στο διαδίκτυο, εργάζονται χωριστά από το πλαίσιο του κύριου εγγράφου. Οι εργαζόμενοι σε υπηρεσίες μπορούν να χειρίζονται ειδοποιήσεις push και να συγχρονίζουν δεδομένα στο παρασκήνιο, να αποθηκεύουν προσωρινή μνήμη ή να ανακτούν αιτήματα πόρων, να υποκλέπτουν αιτήματα δικτύου και να λαμβάνουν κεντρικές ενημερώσεις ανεξάρτητα από το έγγραφο που τα έχει καταχωρίσει, ακόμη και όταν αυτό το έγγραφο δεν είναι φορτωμένο. [26]

Οι service workers περνούν έναν κύκλο ζωής τριών σταδίων εγγραφής, εγκατάστασης και ενεργοποίησης. Η εγγραφή περιλαμβάνει την ενημέρωση του προγράμματος περιήγησης για την τοποθεσία του worker στην υπηρεσία προετοιμασίας για εγκατάσταση. Η εγκατάσταση πραγματοποιείται όταν δεν υπάρχει εγκατεστημένος service worker στο πρόγραμμα περιήγησης για το webapp ή εάν υπάρχει ενημέρωση του service worker. Η ενεργοποίηση πραγματοποιείται όταν όλες οι σελίδες PWA είναι κλειστές, έτσι ώστε να μην υπάρχει σύγκρουση μεταξύ της προηγούμενης έκδοσης και της ενημερωμένης. Ο κύκλος ζωής βοηθά επίσης στη διατήρηση της συνέπειας κατά την εναλλαγή μεταξύ των εκδόσεων του service worker, καθώς μόνο ένας service worker μπορεί να είναι ενεργός για έναν τομέα. [26]

3 React.js

3.1 Εισαγωγή

Το React (γνωστό και ως React.js ή ReactJS) είναι μια δωρεάν βιβλιοθήκη JavaScript ανοιχτού κώδικα ανοιχτού κώδικα [27] για τη δημιουργία διεπαφών χρήστη ή στοιχείων UI. Αναπτύχθηκε και συντηρείται από το Facebook και μια κοινότητα μεμονωμένων προγραμματιστών και εταιρειών. [28] [29] [30] Το React μπορεί να χρησιμοποιηθεί ως βάση στην ανάπτυξη SPA ή εφαρμογών για κινητά. Ωστόσο, το React ασχολείται μόνο με τη διαχείριση της κατάστασης και την απόδοση αυτής της κατάστασης στο DOM, οπότε η δημιουργία εφαρμογών React απαιτεί συνήθως τη χρήση πρόσθετων βιβλιοθηκών για δρομολόγηση, καθώς και ορισμένων λειτουργιών από την πλευρά του πελάτη. [31]

3.2 Ιστορία

Το React δημιουργήθηκε από τον Jordan Walke, έναν μηχανικό λογισμικού στο Facebook, ο οποίος κυκλοφόρησε ένα πρώιμο πρωτότυπο του React που ονομάζεται "FaxJS". [32] [33] Το React Native, το οποίο επιτρέπει την ανάπτυξη native Android, iOS και UWP εφαρμογών με το React, ανακοινώθηκε στο React Conf του Facebook τον Φεβρουάριο του 2015 και ανοιχτού κώδικα τον Μάρτιο του 2015. Στις 18 Απριλίου 2017, το Facebook ανακοίνωσε το React Fiber, ένα νέο σύνολο εσωτερικών αλγορίθμων για απόδοση, σε αντίθεση με τον παλιό αλγόριθμο απόδοσης του React, Stack. [34] Το React Fiber επρόκειτο να γίνει το θεμέλιο κάθε μελλοντικής βελτίωσης και ανάπτυξης χαρακτηριστικών της βιβλιοθήκης React. Η πραγματική σύνταξη για τον προγραμματισμό με το React δεν αλλάζει. Έχει αλλάξει μόνο ο τρόπος εκτέλεσης της σύνταξης. Το παλιό σύστημα απόδοσης του React, το Stack, αναπτύχθηκε σε μια εποχή που η εστίαση του συστήματος στη δυναμική αλλαγή δεν ήταν κατανοητή. Το Stack ήταν αργό για να σχεδιάσει περίπλοκα animations. Το Fiber διασπά το animation σε τμήματα που μπορούν να εξαπλωθούν σε πολλά πλαίσια. Ομοίως, η δομή μιας σελίδας μπορεί να χωριστεί σε τμήματα που μπορούν να διατηρηθούν και να ενημερωθούν ξεχωριστά. Οι λειτουργίες JavaScript και τα εικονικά αντικείμενα DOM ονομάζονται "ίνες" και καθένα μπορεί να λειτουργήσει και να ενημερωθεί ξεχωριστά, επιτρέποντας ομαλότερη απόδοση στην οθόνη. Στις 26 Σεπτεμβρίου του 2017 η έκδοση 16.0 του React κυκλοφόρησε στο κοινό.[35] Στις 16 Φεβρουαρίου 2019, το React κυκλοφόρησε στο κοινό την έκδοση 16.8. [36] Η κυκλοφορία παρουσίασε τα React Hooks. [37] Στις 10 Αυγούστου 2020, η ομάδα της React ανακοίνωσε το React v17.0, αξιοσημείωτο ως η πρώτη σημαντική έκδοση χωρίς σημαντικές αλλαγές στο React developer-facing API. [38]

3.3 Βασική χρήση

Το παρακάτω είναι ένα στοιχειώδες παράδειγμα χρήσης του React σε HTML με JSX και JavaScript.

```
<div id="myReactApp"></div>

<script type="text/babel">
  function Greeter(props) {
    return <h1>{props.greeting}</h1>;
  }
  let App = <Greeter greeting="Hello World!" />;
  ReactDOM.render(App, document.getElementById('myReactApp'));
</script>
```

Η συνάρτηση Greeter είναι ένα React component που δέχεται την ιδιότητα greeting. Η μεταβλητή App είναι μια instance of του στοιχείου Greeter όπου η ιδιότητα greeting έχει οριστεί σε "Hello World!". Στη συνέχεια, η μέθοδος ReactDOM.render καθιστά το στοιχείο Greeter μέσα στο στοιχείο DOM με αναγνωριστικό myReactApp. Όταν εμφανίζεται σε ένα πρόγραμμα περιήγησης ιστού, το αποτέλεσμα θα είναι:

```
<div id="myReactApp">
  <h1>Hello World!</h1>
</div>
```

3.4 Components

Ο κώδικας της React αποτελείται από οντότητες που ονομάζονται components. Τα components μπορούν να αποδοθούν σε ένα συγκεκριμένο στοιχείο στο DOM χρησιμοποιώντας τη βιβλιοθήκη React DOM. Κατά την απόδοση ενός στοιχείου, μπορεί να περαστεί σε τιμές που είναι γνωστές ως "props": [39]

```
ReactDOM.render(<Greeter greeting="Hello World!" />, document.getElementById('myReactApp'));
```

Οι δύο κύριοι τρόποι δήλωσης component στο React είναι μέσω function components και class-based components.

3.5 Function components

Τα function components δηλώνονται με μία μέθοδο όπου επιστρέφει JSX.

```
const Greeter = (props) => <div>Hello, {props.name}!</div>;
```

3.6 Class-based components

Τα class-based components δηλώνονται με την χρήση ES6 κλάσεων.

```
class ParentComponent extends React.Component {
  state = { color: 'green' };
  render() {
    return (
      <ChildComponent color={this.state.color} />
    );
  }
}
```

3.7 Lifecycle μεθόδοι

Οι μεθόδοι lifecycle χρησιμοποιούν μια μορφή hooking που επιτρέπει την εκτέλεση κώδικα σε καθορισμένα σημεία κατά τη διάρκεια ζωής ενός component.

- shouldComponentUpdate: επιτρέπει στον προγραμματιστή να αποτρέψει την άσκοπη εκ νέου απόδοση ενός στοιχείου επιστρέφοντας false εάν δεν απαιτείται απόδοση.
- componentDidMount: καλείται μόλις το στοιχείο έχει "τοποθετηθεί" (το στοιχείο έχει δημιουργηθεί στη διεπαφή χρήστη, συχνά συνδέοντάς το με έναν κόμβο DOM). Αυτό χρησιμοποιείται συνήθως για την ενεργοποίηση της φόρτωσης δεδομένων από απομακρυσμένη πηγή μέσω API.
- componentWillUnmount: καλείται αμέσως πριν το κομμάτι "αποσυναρμολογηθεί". Χρησιμοποιείται συνήθως για την εκκαθάριση των εξαρτήσεων που απαιτούν πόρους από το component που δεν θα αφαιρεθούν απλά με την αποσυναρμολόγηση του στοιχείου (π.χ. κατάργηση οποιωνδήποτε περιπτώσεων setInterval() που σχετίζονται με το component ή ενός "eventListener" που έχει οριστεί στο "document" λόγω της παρουσίας του component)
- render: είναι η πιο σημαντική μέθοδος κύκλου ζωής και η μόνη που απαιτείται σε οποιοδήποτε component. Συνήθως καλείται κάθε φορά που ενημερώνεται η κατάσταση του componen, η οποία θα πρέπει να αντικατοπτρίζεται στη διεπαφή χρήστη.

3.8 JSX

Το JSX ή JavaScript XML, είναι μια επέκταση στη σύνταξη της γλώσσας JavaScript. [40] Παρόμοια σε εμφάνιση με την HTML, το JSX παρέχει έναν τρόπο δομής της απόδοσης συστατικών με σύνταξη που είναι γνωστή σε πολλούς προγραμματιστές. Τα React components γράφονται συνήθως χρησιμοποιώντας JSX, αν και δεν χρειάζεται απαραίτητα (τα components μπορούν επίσης να γραφτούν σε καθαρή JavaScript). Το JSX είναι παρόμοιο με μια άλλη σύνταξη επέκτασης που δημιουργήθηκε από το Facebook για PHP που ονομάζεται XHP.

Παράδειγμα JSX κώδικα:

```
class App extends React.Component {
  render() {
    return (
      <div>
        <p>Header</p>
        <p>Content</p>
        <p>Footer</p>
      </div>
    );
  }
}
```

3.9 Hooks

Τα hooks είναι λειτουργίες που επιτρέπουν στους προγραμματιστές να "γαντζώνονται" σε χαρακτηριστικά React και χαρακτηριστικά του κύκλου ζωής από τα components της συνάρτησης. [41] Τα hooks δεν λειτουργούν μέσα σε classes- σας επιτρέπουν να χρησιμοποιείτε το React χωρίς classes. [42]. Η React παρέχει μερικά built-in hooks όπως το useState, [43] useContext, useReducer και το useEffect. [44] Άλλα τεκμηριώνονται στο Hook API. [45] useState, useReducer και useEffect, που είναι τα πιο χρησιμοποιούμενα, προορίζονται για τον έλεγχο της κατάστασης και των side-effects αντίστοιχα.

3.9.1 Κανόνες των hooks

Υπάρχουν κανόνες [46] που περιγράφουν το χαρακτηριστικό μοτίβο κώδικα στο οποίο βασίζονται τα hooks. Είναι ο σύγχρονος τρόπος χειρισμού της κατάστασης με το React.

1. Τα hooks πρέπει να καλούνται μόνο στο επάνω επίπεδο (όχι μέσα σε βρόχους ή σε if statements).
2. Τα hooks πρέπει να καλούνται μόνο από React function components, όχι από κανονικές συναρτήσεις ή class components

4 AngularJS

4.1 Εισαγωγή

Το AngularJS είναι ένα JavaScript-based ανοιχτού κώδικα web front-end framework για την ανάπτυξη Single Page Applications. Διατηρείται κυρίως από την Google και μια κοινότητα ατόμων και εταιρειών. Στόχος του είναι να απλοποιήσει τόσο την ανάπτυξη όσο και τη δοκιμή τέτοιων εφαρμογών παρέχοντας ένα framework για αρχιτεκτονικές Model-View-Controller (MVC) και Model-View-ViewModel (MVVM), μαζί με components που χρησιμοποιούνται συνήθως σε διαδικτυακές εφαρμογές και Progressive Web Applications.

4.2 Ιστορία

Το AngularJS αναπτύχθηκε αρχικά το 2009 από τον Miško Hevery [46] στην Brat Tech LLC [47] ως το λογισμικό πίσω από μια διαδικτυακή υπηρεσία αποθήκευσης JSON, που θα είχε τιμολογηθεί από το megabyte, για εύκολες εφαρμογές για επιχειρήσεις. Αυτό το εγχείρημα βρισκόταν στον ιστότοπο "GetAngular.com", [47] και είχε λίγους συνδρομητές, πριν αποφασίσουν να εγκαταλείψουν την επιχειρηματική ιδέα και να κυκλοφορήσουν την Angular ως βιβλιοθήκη ανοιχτού κώδικα. Η έκδοση 1.6 πρόσθεσε πολλές από τις έννοιες του Angular στο AngularJS, συμπεριλαμβανομένης της έννοιας της αρχιτεκτονικής εφαρμογών που βασίζεται σε components. [48] Αυτή η έκδοση μεταξύ άλλων αφαιρούσε το Sandbox, το οποίο πολλοί προγραμματιστές πίστευαν ότι παρείχε πρόσθετη ασφάλεια, παρά τις πολυάριθμες ευπάθειες που είχαν ανακαλυφθεί που παρέκαμψαν το sandbox. [49] Η τρέχουσα (από τον Μάρτιο του 2020) σταθερή κυκλοφορία του AngularJS είναι 1.7.9 [50]

4.3 Τεχνικά χαρακτηριστικά

Το AngularJS λειτουργεί διαβάζοντας πρώτα τη σελίδα HTML, η οποία έχει ενσωματωμένα πρόσθετα προσαρμοσμένα χαρακτηριστικά HTML. Η Angular ερμηνεύει αυτά τα χαρακτηριστικά ως οδηγίες για τη σύνδεση τμημάτων εισόδου ή εξόδου της σελίδας σε ένα μοντέλο που αντιπροσωπεύεται από τυπικές μεταβλητές JavaScript. Οι τιμές αυτών των μεταβλητών JavaScript μπορούν να ρυθμιστούν με μη αυτόματο τρόπο στον κώδικα ή να ανακτηθούν από στατικούς ή δυναμικούς πόρους JSON. Το AngularJS βασίζεται στην πεποίθηση ότι ο δηλωτικός προγραμματισμός πρέπει να χρησιμοποιείται για τη δημιουργία διεπαφών χρήστη και τη σύνδεση στοιχείων λογισμικού, ενώ ο επιτακτικός προγραμματισμός ταιριάζει καλύτερα στον καθορισμό της επιχειρηματικής λογικής μιας εφαρμογής. [51] Το framework προσαρμόζει και επεκτείνει την παραδοσιακή HTML για να παρουσιάσει δυναμικό περιεχόμενο μέσω αμφίδρομης δέσμευσης δεδομένων που επιτρέπει τον αυτόματο συγχρονισμό μοντέλων και προβολών. Ως αποτέλεσμα, το AngularJS αποτονίζει τον χειρισμό του DOM με στόχο τη βελτίωση της δοκιμαστικότητας και της απόδοσης. Οι σχεδιαστικοί στόχοι της AngularJS περιλαμβάνουν:

- Αποσύνδεση του χειρισμού DOM από τη λογική της εφαρμογής. Η δυσκολία αυτού επηρεάζεται δραματικά από τον τρόπο δομής του κώδικα.
- Αποσύνδεση της πλευράς του πελάτη μιας εφαρμογής από την πλευρά του διακομιστή. Αυτό επιτρέπει την παράλληλη πρόοδο των εργασιών ανάπτυξης και επιτρέπει την επαναχρησιμοποίηση και των δύο πλευρών.
- Παρέχει δομή για την δημιουργία μιας εφαρμογής: από το σχεδιασμό του UI, μέσω της γραφής της επιχειρησιακής λογικής, έως τη δοκιμή.

Το AngularJS υλοποιεί το μοτίβο MVC για να διαχωρίσει την παρουσίαση, τα δεδομένα και τα λογικά στοιχεία. [52] Χρησιμοποιώντας την εξάρτηση, η Angular φέρνει παραδοσιακά υπηρεσίες διακομιστή, όπως ελεγκτές που εξαρτώνται από την προβολή, σε εφαρμογές ιστού από την πλευρά του πελάτη. Κατά συνέπεια, μεγάλο μέρος του φόρτου στον διακομιστή μπορεί να μειωθεί.

4.4 Αμφίδρομη δέσμευση δεδομένων

Η αμφίδρομη δέσμευση δεδομένων του AngularJS είναι το πιο αξιοσημείωτο χαρακτηριστικό της, απαλλάσσοντας σε μεγάλο βαθμό τον διακομιστή από πρότυπα ευθυνών. Αντ 'αυτού, τα πρότυπα αποδίδονται σε απλό HTML σύμφωνα με τα δεδομένα που περιέχονται σε ένα πεδίο που ορίζεται στο μοντέλο. Η υπηρεσία $scope στο Angular εντοπίζει αλλαγές στην ενότητα μοντέλου και τροποποιεί HTML expressions στην προβολή μέσω ενός controller. Ομοίως, τυχόν αλλαγές στο view αντικατοπτρίζονται στο model. Αυτό παρακάμπτει την ανάγκη ενεργού χειρισμού του DOM και ενθαρρύνει την εκκίνηση και την ταχεία δημιουργία πρωτοτύπων εφαρμογών ιστού. [53] Το AngularJS ανιχνεύει αλλαγές στα μοντέλα συγκρίνοντας τις τρέχουσες τιμές με τις τιμές που είχαν αποθηκευτεί νωρίτερα σε μια διαδικασία βρώμικου ελέγχου, σε αντίθεση με το Ember.js και το Backbone.js που ενεργοποιούν τους listeners όταν αλλάζουν οι τιμές του μοντέλου. [54]

5 Vue.js

5.1 Εισαγωγή

Το Vue.js είναι ένα framework ανοιχτού κώδικα Model-View-ViewModel, front-end, JavaScript για δημιουργία διεπαφών χρήστη και Single Page Applications. Δημιουργήθηκε από τον Evan You και διατηρείται από αυτόν και τα υπόλοιπα ενεργά μέλη της βασικής ομάδας. [55]

5.2 Ιστορία

Το Vue δημιουργήθηκε από τον Evan You αφού εργάστηκε για την Google χρησιμοποιώντας το AngularJS σε πολλά έργα. Αργότερα συνόψισε τη διαδικασία σκέψης του: "Σκέφτηκα, τι θα γινόταν αν μπορούσα απλώς να αποσπάσω το κομμάτι που μου άρεσε πολύ στο Angular και να φτιάξω κάτι πολύ ελαφρύ." [56] Ο πρώτος κώδικας δέσμευσης για το έργο χρονολογήθηκε τον Ιούλιο του 2013 και το Vue κυκλοφόρησε για πρώτη φορά τον επόμενο Φεβρουάριο, το 2014.

5.3 Τεχνικά χαρακτηριστικά

Το Vue.js διαθέτει μια σταδιακά προσαρμόσιμη αρχιτεκτονική που επικεντρώνεται στη δηλωτική απόδοση και τη σύνθεση components. Η βασική βιβλιοθήκη επικεντρώνεται μόνο στο επίπεδο προβολής. [57] Τα προηγμένα χαρακτηριστικά που απαιτούνται για πολύπλοκες εφαρμογές, όπως η δρομολόγηση, η διαχείριση της κατάστασης και τα εργαλεία κατασκευής προσφέρονται μέσω επίσημα συντηρημένων βιβλιοθηκών και πακέτων. [58] Το Vue.js επιτρέπει την επέκταση HTML με χαρακτηριστικά HTML που ονομάζονται directives. [59] Τα directives προσφέρουν λειτουργικότητα σε εφαρμογές HTML και έρχονται είτε ως ενσωματωμένα directives είτε ως καθορισμένα από τον χρήστη.

5.3.1 Components

Τα Vue components επεκτείνουν βασικά στοιχεία HTML για να ενσωματώσουν επαναχρησιμοποιήσιμο κώδικα. Σε υψηλό επίπεδο, τα compoents είναι προσαρμοσμένα στοιχεία στα οποία ο μεταγλωττιστής του Vue αποδίδει συμπεριφορά. Στο Vue, ένα component είναι ουσιαστικά ένα Vue instance με προκαθορισμένες επιλογές. [60] Το απόσπασμα κώδικα παρακάτω περιέχει ένα παράδειγμα ενός Vue component. Το component παρουσιάζει ένα κουμπί και εκτυπώνει τον αριθμό των φορών που γίνεται κλικ στο κουμπί:

```
<template>
  <div id="tuto">
    <button-clicked v-bind:initial-count="0"></button-clicked>
  </div>
</template>

<script>
Vue.component('button-clicked', {
  props: ['initialCount'],
  data: () => ({
    count: 0,
  }),
  template: '<button v-on:click="onClick">Clicked {{ count }} times</button>',
  computed: {
    countTimesTwo() {
      return this.count * 2;
    }
  },
  watch: {
    count(newValue, oldValue) {
      console.log(`The value of count is changed from ${oldValue} to ${newValue}.`);
    }
  },
  methods: {
    onClick() {
      this.count += 1;
    }
  },
  mounted() {
    this.count = this.initialCount;
  }
});

new Vue({
  el: '#tuto',
});
</script>
```

5.3.2 Templates

Το Vue χρησιμοποιεί μια σύνταξη προτύπου βασισμένη σε HTML που επιτρέπει τη σύνδεση του rendered DOM στα Vue instance’s data. Όλα τα πρότυπα Vue είναι έγκυρα HTML που μπορούν να αναλυθούν από προγράμματα περιήγησης συμβατά με τις προδιαγραφές και από αναλυτές HTML. Το Vue συγκεντρώνει τα πρότυπα σε virtual DOM. Ένα εικονικό μοντέλο αντικειμένου εγγράφου ("DOM") επιτρέπει στο Vue να αποδίδει στοιχεία στη μνήμη του πριν ενημερώσει το πρόγραμμα περιήγησης. Σε συνδυασμό με το σύστημα αντιδραστικότητας, το Vue είναι σε θέση να υπολογίσει τον ελάχιστο αριθμό εξαρτημάτων για εκ νέου απόδοση και να εφαρμόσει τον ελάχιστο αριθμό χειρισμών DOM όταν αλλάζει η κατάσταση της εφαρμογής. Οι χρήστες του Vue μπορούν να χρησιμοποιήσουν σύνταξη προτύπου ή να επιλέξουν να γράφουν απευθείας render functions με χρήση υπεργράφου είτε μέσω function calls είτε μέσω JSX. [61] Οι render functions επιτρέπουν την κατασκευή εφαρμογών από στοιχεία λογισμικού. [62]

5.3.3 Reactivity

Το Vue διαθέτει ένα σύστημα αντιδραστικότητας που χρησιμοποιεί απλά αντικείμενα JavaScript και re-rendering. Κάθε component παρακολουθεί τις αντιδραστικές εξαρτήσεις του

κατά τη διάρκεια της απόδοσής του, οπότε το σύστημα γνωρίζει ακριβώς πότε πρέπει να κάνει re-render. [38]

5.3.4 Transitions

Το Vue παρέχει μια ποικιλία τρόπων εφαρμογής των εφέ μετάβασης όταν τα στοιχεία εισάγονται, ενημερώνονται ή αφαιρούνται από το DOM. Αυτό περιλαμβάνει εργαλεία για:

- Αυτόματη εφαρμογή τάξεων για CSS transitions και animations
- Ενσωμάτωση βιβλιοθηκών third-party CSS animation, όπως το Animate.css
- Χρησιμοποιεί JavaScript για να χειριστεί άμεσα το DOM κατά τη διάρκεια των transition hooks
- Ενσωμάτωση thied-party animation βιβλιοθηκών της JavaScript, όπως τοVelocity.js

5.3.5 Routing

Ένα παραδοσιακό μειονέκτημα των SPA είναι η αδυναμία κοινής χρήσης συνδέσμων προς την ακριβή "δευτερεύουσα" σελίδα σε μια συγκεκριμένη ιστοσελίδα. Επειδή τα SPA εξυπηρετούν τους χρήστες τους με μόνο μία απάντηση που βασίζεται σε URL από τον διακομιστή (συνήθως εξυπηρετεί index.html ή index.vue), ο σελιδοδείκτης ορισμένων οθονών ή η κοινή χρήση συνδέσμων σε συγκεκριμένες ενότητες είναι συνήθως δύσκολη αν όχι αδύνατη. Για να λυθεί αυτό το πρόβλημα, πολλοί δρομολογητές από την πλευρά του πελάτη οριοθετούν τις δυναμικές διευθύνσεις URL τους με ένα "hashbang" (#!), Π.χ. page.com/#!/. Ωστόσο, με το HTML5 τα περισσότερα σύγχρονα προγράμματα περιήγησης υποστηρίζουν δρομολόγηση χωρίς hashbangs. Το Vue παρέχει μια διεπαφή για να αλλάξει αυτό που εμφανίζεται στη σελίδα με βάση την τρέχουσα διαδρομή διεύθυνσης URL ανεξάρτητα από τον τρόπο που άλλαξε (είτε μέσω συνδέσμου μέσω ηλεκτρονικού ταχυδρομείου, είτε μέσω ανανέωσης είτε εντός σελίδας). Επιπλέον, η χρήση ενός δρομολογητή front-end επιτρέπει την σκόπιμη μετάβαση της διαδρομής του προγράμματος περιήγησης όταν εμφανίζονται ορισμένα γεγονότα του προγράμματος περιήγησης (δηλ. Κλικ) σε κουμπιά ή συνδέσμους. Το ίδιο το Vue δεν συνοδεύεται από κατακερματισμένη δρομολόγηση front-end. Αλλά το πακέτο ανοιχτού κώδικα "vue-router" παρέχει ένα API για την ενημέρωση της διεύθυνσης URL της εφαρμογής, υποστηρίζει το κουμπί επιστροφής (ιστορικό πλοήγησης) και επαναφέρει τον κωδικό πρόσβασης ηλεκτρονικού ταχυδρομείου ή συνδέσμους επαλήθευσης email με παραμέτρους URL ελέγχου ταυτότητας. Υποστηρίζει τη χαρτογράφηση ένθετων διαδρομών σε ένθετα στοιχεία και προσφέρει λεπτομερή έλεγχο μετάβασης. Με το Vue, οι προγραμματιστές συνθέτουν ήδη εφαρμογές με μικρά δομικά στοιχεία που χτίζουν μεγαλύτερα εξαρτήματα. Με το vue-router να προστίθεται στο μίγμα, τα components πρέπει απλώς να αντιστοιχίζονται στις διαδρομές που ανήκουν και οι διαδρομές γονέα/ρίζας πρέπει να υποδεικνύουν πού πρέπει να κάνουν τα παιδιά. [63]

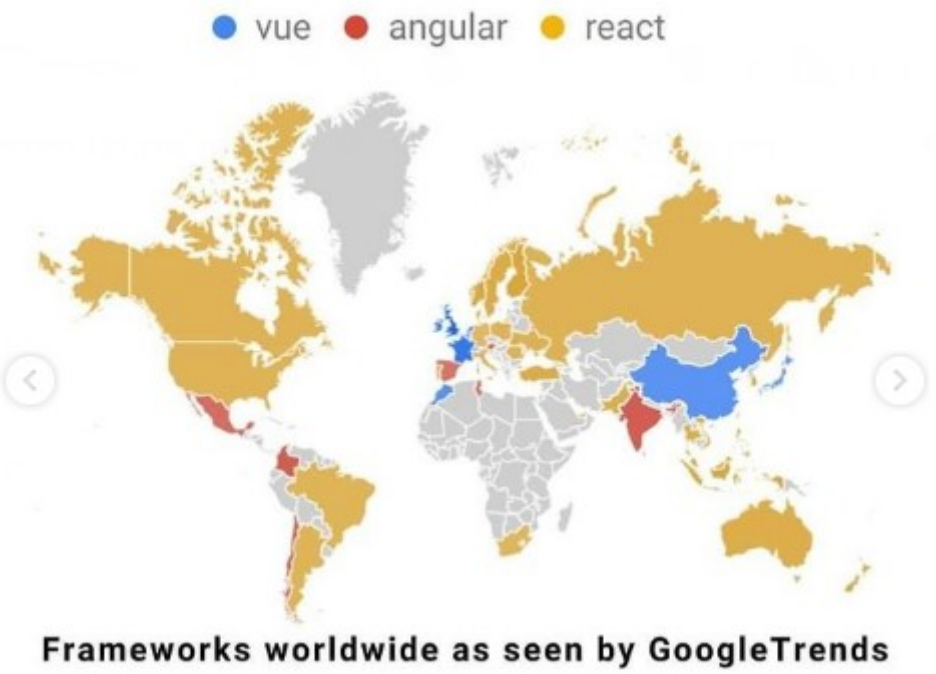
6 React vs Angular vs Vue

6.1 Εισαγωγή

Στο 6ο κεφάλαιο αναλύω τις διαφορές μεταξύ των JavaScript frameworks και τον λογο που με οδηγησε στην επιλογή της React.js.

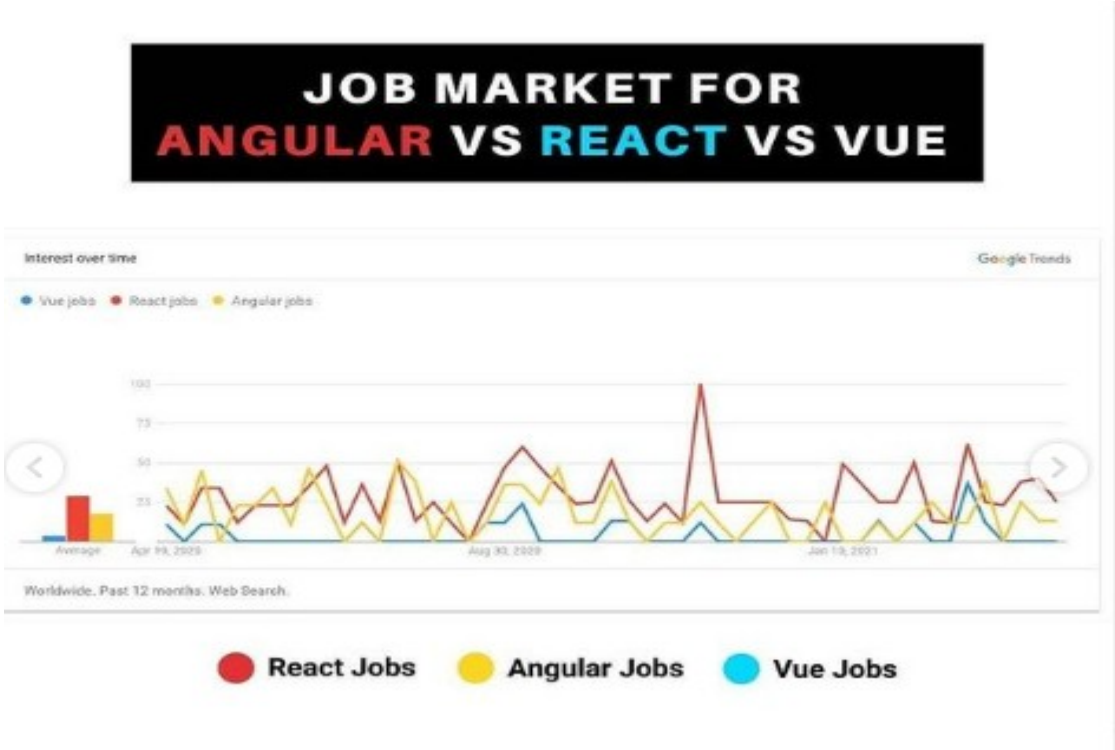
6.2 Σύγκριση

Στο παρακάτω σχήμα βλέπουμε ποιό framework χρησιμοποιούν οι περισσότεροι προγραμματιστές ανα τον κόσμο με βάση το GoogleTrends, όπου πρωτοπόρος φαίνεται να είναι η React. [64]

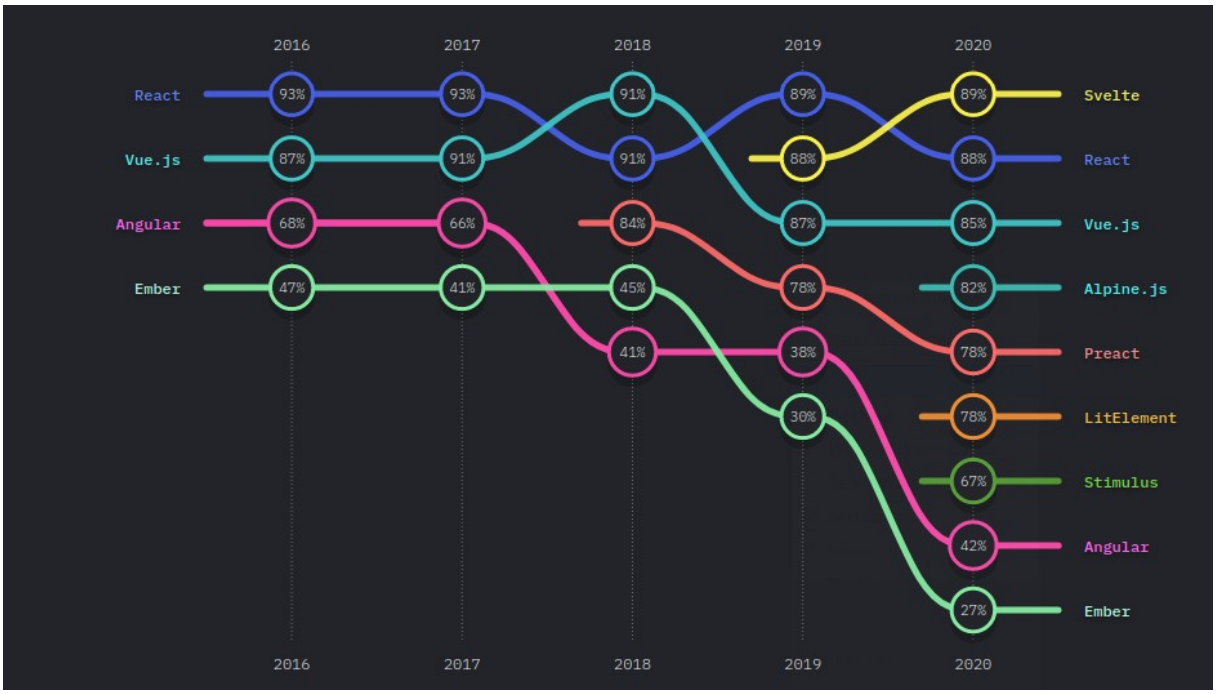


Frameworks worldwide as seen by GoogleTrends

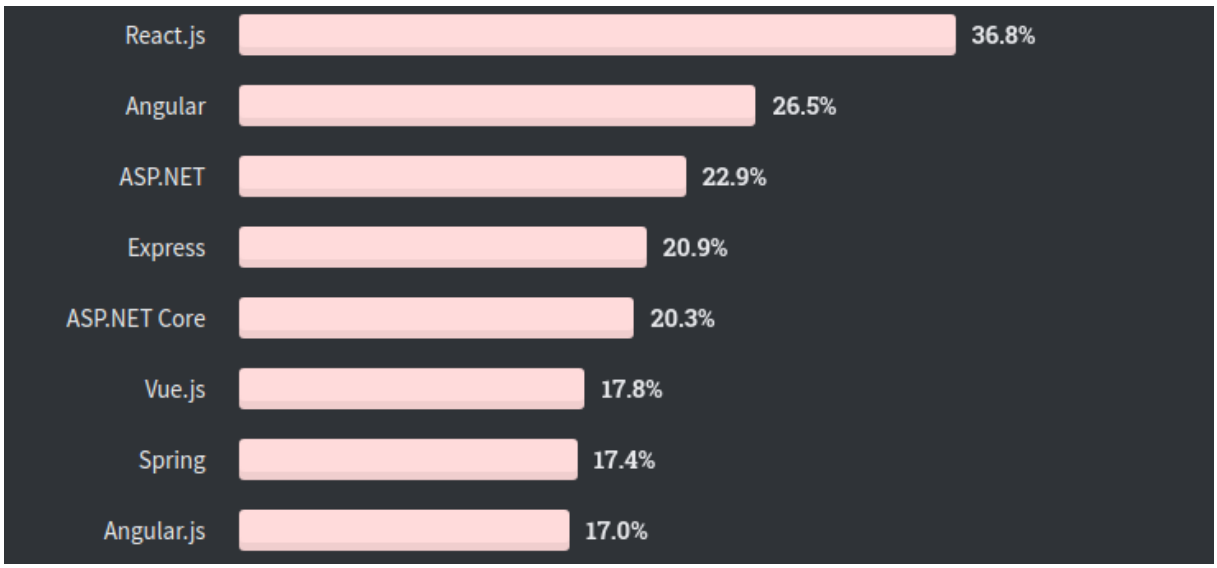
Από το GoogleTrends βλέπουμε επίσης στατιστικά σχετικά με το ποιό framework έχει την μεγαλύτερη ζήτηση στην αγορά εργασίας, όπου η React έχει την μεγαλύτερη ζήτηση τα τελευταία χρόνια. [64]



Τα παρακάτω στατιστικά είναι από το stateofjs.com σχετικά με το τι προτιμούν οι προγραμματστές και βλέπουμε την React να κοντράρετε με το Vue.js, ενώ τον τελευταίο χρόνο την σκητάλη παίρνει το Svelte.js.[65]



Και τέλος, τα στατιστικά απο το insights.stackoverflow.com αναδικνύουν και πάλι πρωτοπόρο την React.[66]



[66]

6.3 Angular: μειονεκτήματα, πλεονεκτήματα και χρήση

Υπάρχουν πολλά οφέλη από το Angular, καθώς είναι ένα JavaScript framework που δημιουργήθηκε για τη δημιουργία εξαιρετικά διαδραστικών εφαρμογών ιστού το Typescript. Είναι ευανάγνωστο, αξιόπιστο, επεκτάσιμο και έχει πολλά χρήσιμα στοιχεία που κάνουν τη διαδικασία ανάπτυξης ακόμη πιο εύκολη. Τα σημαντικότερα πλεονεκτήματα του Angular είναι τα εξής:

- Υποστήριξη Typescript
- MVVM (Model-View-ViewModel)
- Πολυάριθμες προηγμένες δυνατότητες και βιβλιοθήκες.
- Καθαρό και λεπτομερές documentation

Το Angular είναι ενα framework που λειτουργεί καλά για αντικειμενοστραφή προγραμματισμό. Κυρίως, το Angular είναι κατάλληλο για την ανάπτυξη real-time εφαρμογών, native, υβριδικών και διαδικτυακών εφαρμογών. Ωστόσο, όπως κάθε άλλο framework ανάπτυξης λογισμικού, το Angular δεν είναι χωρίς μειονεκτήματα. Λάβετε υπόψη τα ακόλουθα στοιχεία σχετικά με την Angular όταν επιλέγετε ένα framework:

- Το Angular είναι δύσκολο να διαβαστεί σε σύγκριση με το React ή το Vue λόγω των πολλών δομικών στοιχείων, όπως components, modules και pipes.
- Αργή απόδοση σε σύγκριση με άλλα framework και αργή διαδικασία rendering.

6.4 React: μειονεκτήματα, πλεονεκτήματα και χρήση

Ποια είναι τα βασικά οφέλη της React; Η React χρησιμοποιείται ευρέως για την ανάπτυξη εφαρμογών μεταξύ πλατφορμών καθώς και την ανάπτυξη SPA. Επιπλέον, η React προσφέρει μεγάλη κοινοτική υποστήριξη καθώς είναι το πιο δημοφιλές framework στον κόσμο. Είναι ευέλικτο, ασφαλές και σας επιτρέπει να δημιουργείτε γρήγορα ελαφριές εφαρμογές. Υπάρχουν πολλά πλεονεκτήματα της React:

- Εύκολη σύνταξη JSX για πρότυπα.
- Υψηλή ταχύτητα επεξεργασίας λόγω virtual DOM.
- Υψηλής ποιότητας documentation.
- Ευρεία επαγγελματική κοινότητα και πολλές έτοιμες λύσεις.
- Χαρακτηριστικά μονόδρομης δέσμευσης δεδομένων.
- Ο κώδικας που δημιουργείται με την React είναι επαναχρησιμοποιήσιμος και εύκολος στη δοκιμή.
- Εύκολη μετάβαση στις πιο πρόσφατες εκδόσεις του πλαισίου ανάπτυξης.

Δεν υπάρχουν πολλά μειονεκτήματα της React. Ειδικότερα, αυτό το πλαίσιο ενημερώνεται τόσο συχνά που δεν μπορείτε πάντα να βρείτε την απαραίτητη τεχνική τεκμηρίωση εγκαίρως. Η React είναι επίσης τόσο ευέλικτη και πολυλειτουργική που έχει γίνει πραγματικό πρόβλημα για τους άπειρους προγραμματιστές, καθώς και για όσους εργάζονται σε μια ομάδα καθώς το ίδιο πρόβλημα μπορεί να λυθεί με πολλούς διαφορετικούς τρόπους. Άλλα μειονεκτήματα της React:

- Χωρισμός κοινότητας μεταξύ εκείνων που τηρούν τα παραδοσιακά φύλλα στυλ CSS Modules και των οπαδών της χρήσης Emotion και Styled Components CSS-in-JS.
- Η React άφησε τον αντικειμενοστραφή προγραμματισμό, τον οποίο έχουν συνηθίσει πολλοί προγραμματιστές.
- Το JSX μπορεί να φαίνεται μπερδεμένο εάν τα πρότυπα είναι πολλά και τέμνονται.

6.5 Vue: μειονεκτήματα, πλεονεκτήματα και χρήση

Εν ολίγοις, το Vue επιτρέπει να δημιουργήσετε διάφορες προσαρμόσιμες εφαρμογές υψηλής απόδοσης και να εισέλθετε πιο γρήγορα στην αγορά. Αυτό το εργαλείο είναι ιδανικό για μια μικρή και ελαφριά SPA,. Όλα τα πλεονεκτήματα του Vue συνδέονται στενά με τη μοναδική του φύση: αυτό το framework δημιουργήθηκε από μία κοινότητα προγραμματιστών με μεγάλη εμπειρία στο React και το Angular. Για το λόγο αυτό, το Vue δεν περιέχει τα μειονεκτήματα αυτών των δύο συστημάτων και, ταυτόχρονα, έχει παρόμοια χαρακτηριστικά με αυτά.

Ωστόσο το Vue.js είναι ένα πρόσφατο φαινόμενο. Αυτός είναι ο λόγος για τον οποίο υπάρχουν πολύ λίγες πηγές πληροφοριών για την επίλυση προβληματικών ζητημάτων (σε σύγκριση με την React

και το Angular) με το Vue, καθώς και υπάρχει ανεπαρκής αριθμός περιπτώσεων για να μάθουν οι προγραμματιστές.

7 Web Storage API

7.1 Εισαγωγή

Με το web storage, οι εφαρμογές ιστού μπορούν να αποθηκεύουν δεδομένα τοπικά στο πρόγραμμα περιήγησης του χρήστη. Πριν από την HTML5, τα δεδομένα της εφαρμογής έπρεπε να αποθηκευτούν σε cookie, που περιλαμβάνονται σε κάθε αίτημα διακομιστή. Το web storage είναι πιο ασφαλής και μεγάλα ποσά δεδομένων μπορούν να αποθηκευτούν τοπικά, χωρίς να επηρεαστούν οι επιδόσεις του ιστότοπου. Σε αντίθεση με τα cookie, το όριο αποθήκευσης είναι πολύ μεγαλύτερο (τουλάχιστον 5MB) και οι πληροφορίες δεν μεταφέρονται ποτέ στον διακομιστή. Το web storage είναι ανά προέλευση (ανά τομέα και πρωτόκολλο). Όλες οι σελίδες, από μία προέλευση, μπορούν να αποθηκεύουν και να έχουν πρόσβαση στα ίδια δεδομένα.[67]

7.2 Web Storage Objects

Το HTML web storage παρέχει δύο αντικείμενα για την αποθήκευση δεδομένων στον πελάτη:

- localStorage – όπου αποθηκεύει δεδομένα χωρίς ημερομηνία λήξης
- sessionStorage – όπου τα δεδομένα χάνονται μόλις κλείσει ο browser

7.3 localStorage

Το localStorage αποθηκεύει τα δεδομένα χωρίς ημερομηνία λήξης. Τα δεδομένα δεν θα διαγραφούν όταν το πρόγραμμα περιήγησης κλείσει και θα είναι διαθέσιμα την επόμενη ημέρα, εβδομάδα ή έτος.

//examples

```
localStorage.setItem("token",token);
localStorage.setItem("user",JSON.stringify(res.data));

const token = localStorage.getItem("token");
const user = JSON.parse(localStorage.getItem("user")) || [];
```

7.4 sessionStorage

Το αντικείμενο sessionStorage είναι ίσο με το αντικείμενο localStorage, εκτός από το ότι αποθηκεύει τα δεδομένα μόνο για μία περίοδο σύνδεσης. Τα δεδομένα διαγράφονται όταν ο χρήστης κλείσει τη συγκεκριμένη καρτέλα του προγράμματος περιήγησης

8 PayPal και integration με React.js

8.1 Εισαγωγή

Το PayPal είναι πύλη ηλεκτρονικών πληρωμών μέσω της οποίας διεκπεραιώνονται μεταφορές χρημάτων και πληρωμές και ανήκει στην αμερικανική εταιρεία PayPal Holdins, inc. Το PayPal εκτελεί την επεξεργασία των πληρωμών για online πωλήσεις, δημοπρασίες χώρων, καθώς και άλλους εμπορικούς χρήστες, για την οποία χρεώνει αμοιβή. Φορτίζει μερικές φορές επίσης τέλος συναλλαγής για τη λήψη χρημάτων (ένα ποσοστό του ποσού που απέστειλε συν ένα πρόσθετο σταθερό ποσό). Το επίπεδο των τελών εξαρτάται από το χρησιμοποιούμενο νόμισμα, την επιλογή πληρωμής που χρησιμοποιείται, τη χώρα του αποστολέα, τη χώρα του δικαιούχου, το ποσό που αποστέλλεται και τον τύπο του λογαριασμού του δικαιούχου.[74]

8.2 Ασφάλεια

Στην προσπάθεια του να αυξήσει την ασφάλεια των χρηστών του, το PayPal δημιούργησε μια δύο σταδίων διαδικασία σύνδεσης (log-in). Ο χρήστης αν το επιθυμεί μπορεί πέραν του να χρησιμοποιεί το username του και το password του να χρησιμοποιήσει επίσης το PayPal security key. Ουσιαστικά ο χρήστης αφού εισάγει το username και το password του οδηγείται σε μία σελίδα στην οποία πρέπει να συμπληρώσει το security key. Αφού το κάνει αυτό πλέον έχει πρόσβαση στο λογαριασμό του. Στην περίπτωση που ο χρήστης έχει επιλέξει να χρησιμοποιεί το security key αλλά δεν το έχει μαζί του είτε το έχει χάσει , στην σελίδα που κανονικά θα συμπλήρωνε τον αριθμό του security key μπορεί να επιλέξει τον σύνδεσμο lost or forgotten security key. Εκεί θα κληθεί να απαντήσει σε ερωτήσεις ασφαλείας ώστε να πιστοποιήσει την αυθεντικότητα του.
Το security key είναι μια πολύ μικρή συσκευή στο μέγεθος ενός USB stick και φέρει επάνω του μια μικρή οθόνη υγρών κρυστάλλων και ένα μικρό κουμπί. Κάθε φορά που ο χρήστης προσπαθεί να κάνει log-in στον λογαριασμό του, όταν του ζητηθεί το security key αυτός πιέζει το κουμπί πάνω στη συσκευή και γράφει στη σελίδα τον εξαψήφιο κωδικό που του εμφανίστηκε στην οθόνη της συσκευής. Την συσκευή αυτή μπορεί να την αγοράσει κανείς από την ιστοσελίδα του PayPal. Το κόστος της είναι 5$ και αυτή θα φτάσει στον αγοραστή μέσω ταχυδρομείου. Όταν ο χρήστης παραλάβει την συσκευή μπορεί να την ενεργοποιήσει στην ιστοσελίδα του PayPal εισάγοντας τον σειριακό αριθμό που θα βρει στο πίσω μέρος της συσκευής.

Σημαντικό είναι να αναφέρουμε πως το security key μπορούν μέχρι τώρα να το χρησιμοποιήσουν μόνο χρήστες που είναι εγκατεστημένοι στην Αυστραλία, την Γερμανία, τον Καναδά, το Ηνωμένο Βασίλειο και στις Ηνωμένες Πολιτείες Αμερικής.

Ένας δεύτερος τρόπος για να αυξήσει ο χρήστης την ασφάλεια του λογαριασμού του είναι το MTan (Mobile Transaction Authentication Number). Ουσιαστικά κάθε φορά που ο χρήστης προσπαθεί να πραγματοποιήσει μια συναλλαγή του έρχεται στο κινητό του τηλέφωνο γραπτό μήνυμα από το PayPal το οποίο περιέχει στοιχεία της συναλλαγής καθώς και ένα password. Το password αυτό ο

χρήστης πρέπει να το επιβεβαιώσει στη σελίδα του PayPal ώστε η συναλλαγή να ολοκληρωθεί. Ωστόσο η ασφάλεια αυτής της τεχνικής εξαρτάται πάντα από την ασφάλεια του συστήματος της κινητής τηλεφωνίας που υπάρχει. Η χρήση του MTan δεν έχει καμία οικονομική επιβάρυνση, εκτός από τις καθιερωμένες χρεώσεις γραπτών μηνυμάτων που ο φορέας παροχής της κινητής σύνδεσης επιβάλει.[74]

8.3 Integration με React.js

Οι προγραμματιστές που ενσωματώνουν το PayPal αναμένεται να προσθέσουν το JS SDK <script> σε έναν ιστότοπο και στη συνέχεια να αποδώσουν στοιχεία όπως τα κουμπιά PayPal μετά τη φόρτωση του σεναρίου. Αυτή η αρχιτεκτονική λειτουργεί για απλούς ιστότοπους, αλλά μπορεί να είναι προκλητική κατά τη δημιουργία SPA. Οι προγραμματιστές React σκέφτονται ως προς τα components και όχι για τη φόρτωση εξωτερικών script από ένα αρχείο index.html. Είναι εύκολο να καταλήξετε σε μια ενσωμάτωση React PayPal που δεν είναι βέλτιστη και βλάπτει την εμπειρία χρήστη του αγοραστή. Για παράδειγμα, η αφαίρεση όλων των λεπτομερειών υλοποίησης των κουμπιών PayPal σε ένα μόνο στοιχείο React είναι ένα αντι-μοτίβο επειδή συνδυάζει στενά τη φόρτωση του σεναρίου με την απόδοση. Είναι επίσης προβληματικό όταν πρέπει να αποδώσετε πολλά διαφορετικά στοιχεία PayPal που μοιράζονται τις ίδιες global script παραμετρους.

Το react-paypal-js παρέχει μια λύση στους προγραμματιστές για να αφαιρέσουν τις πολυπλοκότητες γύρω από τη φόρτωση του JS SDK. Επιβάλλει τις βέλτιστες πρακτικές από προεπιλογή, ώστε οι αγοραστές να έχουν την καλύτερη δυνατή εμπειρία χρήστη.

Χαρακτηριστικά:

- Επιβολή ασύγχρονου φορτώματος του JS SDK εκ των προτέρων, οπότε όταν έρθει η ώρα να αποδώσετε τα κουμπιά στον αγοραστή σας, θα εμφανιστούν αμέσως.
- Αφαίρεση της πολυπλοκότητας της φόρτωσης του SDK JS με το global component PayPalScriptProvider.
- Υποστήριξη ενεργειών αποστολής για επαναφόρτωση του JS SDK και εκ νέου απόδοση στοιχείων όταν αλλάζουν global παράμετροι, όπως το νόμισμα.
- Εύκολα στη χρήση εξαρτήματα για όλες τις διαφορετικές προσφορές προϊόντων Braintree/PayPal

9 Thessaloniki club

9.1 Εισαγωγή

Ως αποτέλεσμα της μελέτης μου, υλοποίησα μια Single Page Application με την ονομασία Thessaloniki club. Η εφαρμογή παρουσιάζει προσφορές απο μαγαζιά της Θεσσαλονίκης, και αλληλεπιδρά με ενα API για κρατήσεις και πληρωμές

9.2 Registration

Η διαδικασία του registration γίνεται στέλνοντας τις πληροφορίες του χρήστη στο API, αυτό με την σειρά του δημιουργει τον χρήστη μαζί με ενα μοναδικό user token, όπου θα χρησιμοποιείται για την περιήγησή στην εφαρμογή. Οι πληροφορίες για τον χρήστη αποθηκεύονται στο local storage.

```
axios.post(apiRoot+"api/Members/Register",
{
    "firstName": document.getElementById("firstname").value,
    "lastName": document.getElementById("lastname").value,
    "email": document.getElementById("email").value,
    "password": document.getElementById("pass").value,
    "address": document.getElementById("address").value,
    "city": document.getElementById("city").value,
    "zipCode": document.getElementById("zipcode").value,
    "phoneNumber": document.getElementById("phone").value,
    "sex": parseInt(document.getElementById("gender").value) == 0 || parseInt(document.getElementById("gender").value) == 1 ?
    parseInt(document.getElementById("gender").value) : null,
    "deviceName": DeviceModel || "",
    "deviceId": Android_id || "",
    "deviceType": dType || 0
})
    .then((response) => {
    //console.log(response.data.token);
        const api = apiRoot+'api/Members/GetMemberProfile';
        const token = response.data.token;

        if(window.AndroidFunction){
            window.AndroidFunction.SaveToken(token);
        }

        if(dType && dType == 1){
            window.webkit.messageHandlers.saveUserToken.postMessage(JSON.stringify({token}));
        }

        localStorage.setItem("token",token);
        axios.get(api , { headers: {"Authorization" : `Bearer ${token}`} })
        .then(res => {
        //console.log(res);
            localStorage.removeItem("user");
            localStorage.setItem("user",JSON.stringify(res.data));
            //var user = JSON.parse(localStorage.getItem("user")) || [];
            //console.log(user);


        });
    }, (error) => {
    console.log(error);
});
```

9.3 Login

Η διαδικασία του login γίνεται στέλνοντας τις πληροφορίες του χρήστη στο API, αυτό με την σειρά του ψάχνει στην βάση για να βρεί το email και το password του χρήστη για την ταυτοποίηση. Αφού γίνει επιτυχής ταυτοποίηση, επιστρέφει το user token που δημιουργήθηκε κατά το registration

του χρήστη, το οποίο χρησιμοποιείται για την περιήγησή στην εφαρμογή. Οι πληροφορίες του χρήστη αποθηκεύονται στο local storage.

```
axios.post(apiRoot+"api/Members/login",
{
    "email": mail,
    "password": document.getElementById("pass").value
})
.then((response) => {
    //console.log(response.data.token);
        const api = apiRoot+'api/Members/GetMemberProfile';
        const token = response.data.token;
        if(window.AndroidFunction){
            window.AndroidFunction.SaveToken(token);
        }
        if(dType && dType == 1){
            window.webkit.messageHandlers.saveUserToken.postMessage(JSON.stringify({token}));
        }
        localStorage.setItem("token",token);

        //get user
        axios.get(api , { headers: {"Authorization" : `Bearer ${token}`} })
        .then(res => {
        //console.log(res);
            //localStorage.removeItem("user");
            localStorage.setItem("user",JSON.stringify(res.data));
            //console.log(user);
            history.push("/Welcome");

        });

}, (error) => {
    setWrong(true);
});
```

9.4 Forgot password

Στην εφαρμογή υπάρχει υλοποιημένη και διαδικασία επαναφοράς κωδικού, όπου καλώντας το API και παιρνώντας το mail του χρήστη, αποστέλλεται η διαδικασία επαναφοράς.

```
function sendMail(){
    axios.post(apiRoot+'api/Members/ForgotPassword?email='+document.getElementById('mail').value+'')
    .then(res => {
        console.log(res);
        if(res.status==200){
            document.getElementById("res").innerHTML = res.data;
            setTimeout(() => {
                history.push('/SignIn');
            }, 3000)
        }
    }, (error) => {
        console.log(error);
    });
}
```

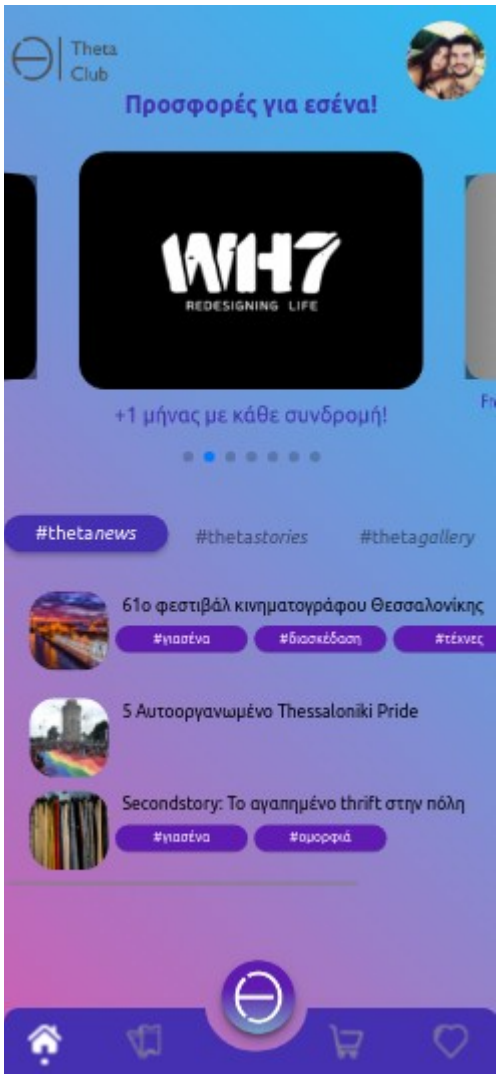
9.5 Προσθήκη στο καλάθι

Η διαδικασία της προσθήκης στο καλάθι γίνεται ξεχωριστά για κάθε προϊόν στο API. Αποθηκεύονται όλα τα προϊόντα που πρόσθεσε ο χρήστης στο καλάθι σε έναν πίνακα στο local storage, και με loop του πίνακα, στέλνονται ανά προϊόν τα δεδομένα στο API

```
cart.map(item => {
    axios.post(api , {
        "id": 0,
        "creationDate": "2021-08-14T10:43:38.541Z",
        "offerId": item.id,
        "status": 0,
        "totalPrice": item.price,
        "couponURL": "string",
        "orderLines": [
        {
            "id": 0,
            "creationDate": "2021-08-14T10:43:38.541Z",
            "productId": item.product,
            "quantity": 0,
            "price": item.price
        }
        ]
    },{ headers: {"Authorization" : `Bearer ${token}`} } )
    .then(res => {
        console.log(res);
        if(res.status==200){
            totalPrice += item.price;
            ids.push(
                res.data
            );
            if(totalPrice>0){
                history.push({
                    pathname: "/BuyComplete",
                    state:{
                        value: totalPrice,
                        ids: ids
                    }
                });
            }else{
                history.push("/BuyCompleteFree");
            }
```

9.6 Slider προσφορών με την χρήση της βιβλιοθήκης swiper/react

Με την χρήση της βιβλιοθήκης swiper/react γίνεται render στην σελίδα ενα slider το οποίο τραβάει τις προσφορές απο το API και την εμφανίζει.




```
//libraries
import { Swiper, SwiperSlide } from 'swiper/react';
import SwiperCore, {
  EffectCoverflow,Pagination
} from 'swiper/core';
import { BrowserRouter as Router, Switch, Route, NavLink, Link } from 'react-router-dom'
import * as axios from 'axios';
import React, {Component,useState,useEffect} from 'react';


// Import Swiper styles
import "swiper/swiper.min.css";
import "swiper/components/effect-coverflow/effect-coverflow.min.css"
import "swiper/components/pagination/pagination.min.css"

SwiperCore.use([EffectCoverflow,Pagination]);
```

Κάθε slide είναι και ένας σύνδεσμος ο οποιός οδηγεί σε μία νέα σελίδα η οποία περιέχει πληροφορίες για την προσφορά καθώς και διαθέτει επιλογές για προσθήκη στο καλάθι η στην wishlist.

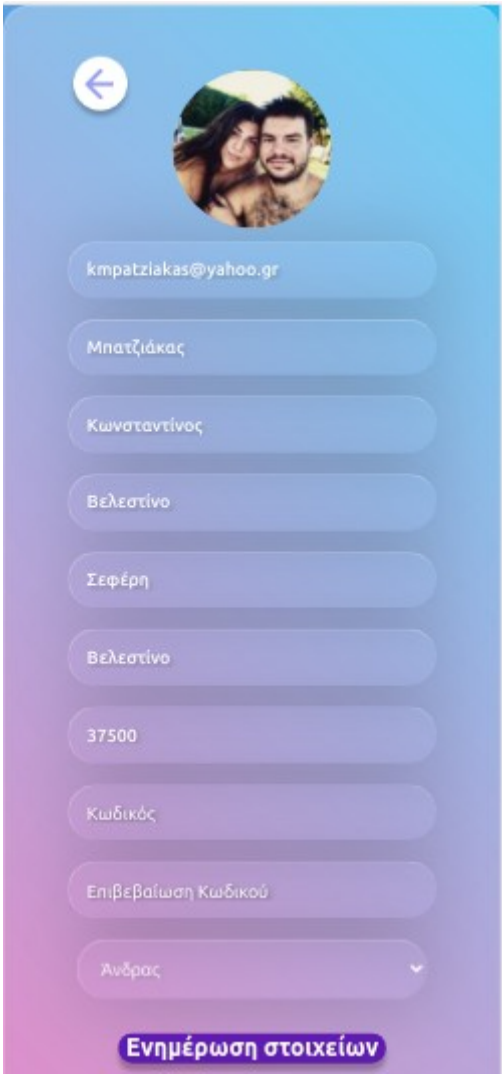
```
//getOffers
axios.get(apiRoot+'api/Offers/GetOffers', { headers: {"Authorization" : `Bearer ${token}`} })
.then(res => {
    //console.log(res.data);
    localStorage.setItem("offers",JSON.stringify(res.data));
});

const offers = JSON.parse(localStorage.getItem("offers")) || [];

  return (
    <div>
      <h3 className="new-offers-h1">Προσφορές για εσένα!</h3>
      <div className="main-slides">
        <Swiper effect={'coverflow'} grabCursor={true} centeredSlides={true} slidesPerView={'auto'} spaceBetween={10} coverflowEffect={{
          "rotate": 50,
          "stretch": 0,
          "depth": 100,
          "modifier": 1,
          "slideShadows": true
        }} pagination={true} className="mySwiper">
          {offers.map((offer) => (
          <SwiperSlide key={offer.id} style={{"backgroundImage":"url("+b2bRoot+offer.photoURL+")"}}>
            <div className="carousel-offer-text">{offer.name}</div>
            <NavLink className="slider-link" to={{
            pathname: "/Offer1",
            state: {
              id:offer.id,
              image: offer.photoURL,
              header: offer.name,
              par:offer.description,
              address: offer.location,
              product:offer.products[0].id,
              price:offer.value,
              index:offers.indexOf(offer),
            },
          }}></NavLink></SwiperSlide>
          ))}
        </Swiper>
      </div>
    </div>
  )
}
```

9.7 Προφίλ χρήστη και ενημέρωση στοιχείων του

Ο χρήστης έχει την δυνατότητα να δει και να ενημερώσει τα στοιχεία που έδωσε κατά την διαδικασία του registration, όπως email,password, Όνομα κ.α. Επίσης έχει την δυνατότητα να ορίσει και φωτογραφία προφίλ. Η ενημέρωση στοιχείων γίνεται με 2 κλήσεις στο API, μία για τις προσωπικές πληροφορίες του, και μία για το ανέβασμα φωτογραφίας



Με την χρήση του createObjectURL δημιουργείται ένα DOMString που περιέχει μια διεύθυνση URL που αντιπροσωπεύει το αντικείμενο που δίνεται στην παράμετρο.

```
function onImageChange(event){
    console.log(event.target.files[0]);
    if (event.target.files && event.target.files[0]) {
        console.log(event.target.files[0].name);

        image = URL.createObjectURL(event.target.files[0]);
        file = event.target.files[0];
        name = event.target.files[0].name;
        document.getElementById('profile-pic').style.backgroundImage = "url("+image+")";
    }
};
```

Η φωτογραφία που θέλει ο χρήστης να ορίσει ως φωτογραφία προφίλ στέλνεται με την μορφή FormData, όπου περιέχει το αρχείο και το όνομα της φωτογραφίας.

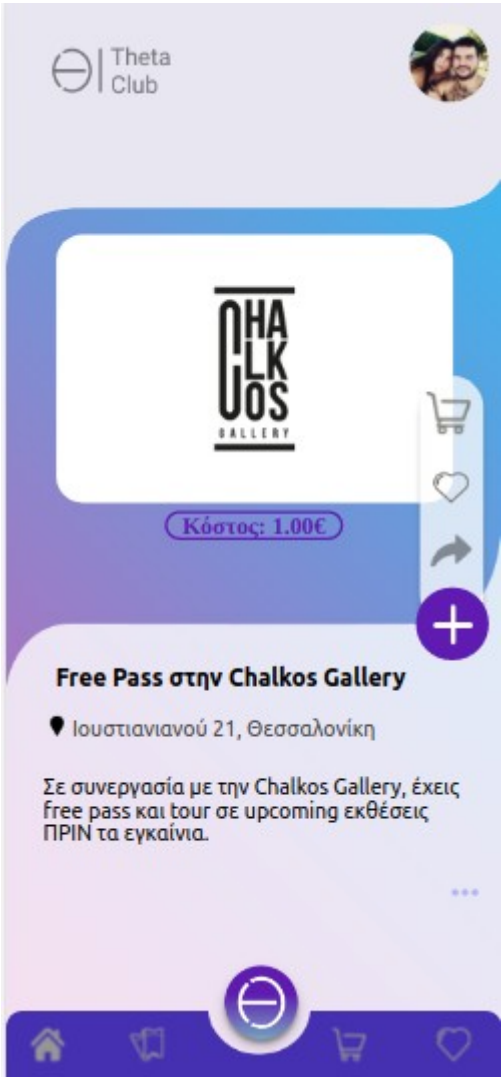
```
const formData = new FormData();
formData.append("FileName",name);
formData.append("ImageFile",file);

if(file != null && name != null){
    axios.post(apiRoot+'api/Members/UpdateUserImage', formData
    , { headers: {"Authorization" : `Bearer ${token}`, "Content-Type":"multipart/form-data"} })
    .then(res => {
        console.log(res);
        history.push("/AfterUpload");
    }, (error) => {
        console.log(error);
    });
}

axios.post(apiRoot+'api/Members/UpdateUserInfo', {
    "firstName": firstName,
    "lastName": lastName,
    "email": mail,
    "password": pass != null && pass == confirm ? pass : "",
    "address": address,
    "city": city,
    "zipCode": zipCode,
    "phoneNumber": phoneNumber,
    "sex": sex,
    "deviceName": user.deviceName,
    "deviceId": user.deviceId,
    "deviceType": user.deviceType,
    "imageFile": file,
    "fileName": name
}
, { headers: {"Authorization" : `Bearer ${token}`} })
.then(res => {
    console.log(res);
    //get user
    const token = localStorage.getItem("token");
    const api = apiRoot+'api/Members/GetMemberProfile';
    axios.get(api , { headers: {"Authorization" : `Bearer ${token}`} })
    .then(res => {
        //console.log(res);
        //localStorage.removeItem("user");
        localStorage.setItem("user",JSON.stringify(res.data));
        //console.log(user);
        history.push("/AfterUpload");
    });
    user = JSON.parse(localStorage.getItem("user")) || [];
```

9.8 Προβολή προσφοράς

Η προβολή πληροφοριών μίας προσφοράς γίνεται μέσω της διαδικασίας αποστολής παραμέτρων απο σελίδα σε σελίδα. Όταν γίνεται render το slider με τις προσφορές δημιουργούνται σύνδεσμοι όπου περιέχουν όλες τις πληροφορίες της προσφοράς και με κλικ πάνω στον σύνδεσμο φορτώνει η σελίδα με τις πληροφορίες της προσφοράς. Σε αυτή την σελίδα δίνονται και οι δυνατότητες προσθήκης της προσφοράς στην wishlist και στο καλάθι.



```
const { handle } = useParams();
const location = useLocation();
const { image } = location.state;
const { header } = location.state;
const { par } = location.state;
const { address } = location.state;
const { id } = location.state;
const { product } = location.state;
const { price } = location.state;
```

Η Μέθοδος για την προσθήλη στο καλάθι, τα προϊόντα αποθηκεύονται σε έναν array στο local storage. Με την ίδια λογική ακριβώς γίνεται και η προσθήκη στην wishlist.

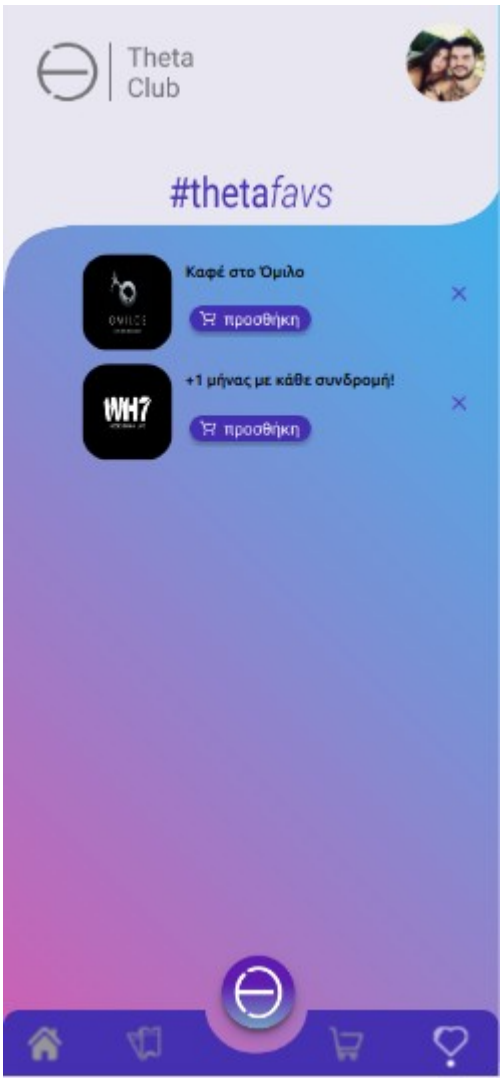
```
const addCart = () => {
    var quantity = 1;
    cart.map( (item) => (
        item.id==id ? quantity++ : ''
    ))
    axios.get(apiRoot+'api/Offers/CheckMemberPermittedQuantity/?OfferId='+id+'&Quantity='+quantity+'' ,
    { headers: {"Authorization" : `Bearer ${token}`} })
    .then(res => {
        console.log(res);
        if(res.data === ""){
            cart.push({
                id:id,
                img:image,
                price:price,
                info:header,
                product:product,
            });
            //localStorage.removeItem("cart");
            localStorage.setItem("cart",JSON.stringify(cart));
            setAdd(!add);
        }else{
            setAddFalse(!addFalse);
        }
        //document.getElementById("fault").innerHTML = JSON.stringify(res);
    } , (error) => {
        console.log(error);
    });
}
```

```
const addWishlist = () => {
    var quantity = 1;
    cart.map( (item) => (
        item.id==id ? quantity++ : ''
    ))
    const api = apiRoot+'api/WishLists/Add/?OfferId='+id+'';
    axios.get(api , { headers: {"Authorization" : `Bearer ${token}`} })
    .then(res => {
        console.log(res);
        if(res.data === ""){
            axios.get(apiRoot+'api/Offers/CheckMemberPermittedQuantity/?OfferId='+id+'&Quantity='+quantity+'' ,
            { headers: {"Authorization" : `Bearer ${token}`} })
            .then(res => {
                console.log(res);
                if(res.data === ""){
                    wishlist.push({
                        id:id,
                        img:image,
                        price:price,
                        info:header,
                        product:product
                    });
                    //localStorage.removeItem("cart");
                    localStorage.setItem("wishlist",JSON.stringify(wishlist));
                    setWish(!wish);
                }else{
                    setWishFalse(!wishFalse);
                }
            });
        }else{
            setWishFalse(!wishFalse);
        }
        //document.getElementById("fault").innerHTML = JSON.stringify(res);
    }, (error) => {
        //document.getElementById("fault").innerHTML = error + "" + token;
    });


}
```

9.9 Wishlist

Εδώ ο χρήστης μπορεί να δει και να επεξεργαστεί τις προσφορές της wishlist του. Υπάρχουν διαθέσιμες οι επιλογές διαγραφής μιας προσφοράς και η προσθήκη της στο καλάθι.



Η μέθοδος διαγραφής από την wishlist γίνεται με βάση το id της προσφοράς, ο αλγόριθμος ψάχνει στο array να βρει το id, αν το βρει αφαιρεί την συγκεκριμένη προσφορά, δημιουργει καινούριο array και τον περνάει στην θέση του παλιού array στο local storage

```
const del = (e,id) => {
    var newWishlist = JSON.parse(localStorage.getItem("wishlist")) || [];
    localStorage.removeItem("wishlist");
    localStorage.setItem("wishlist",JSON.stringify(newWishlist.filter((item) => item.id !== id)));
    wishlist = JSON.parse(localStorage.getItem("wishlist")) || [];
    e.target.parentNode.remove();
    const api = apiRoot+'api/WishLists/Remove/?OfferId='+id+'';
    axios.get(api , { headers: {"Authorization" : `Bearer ${token}`} })
    .then(res => {
        console.log(res);
    });
    if(wishlist.length == 0){
        var div = document.createElement("div");
        div.className = "empty-wishlist";
        div.innerHTML = "Η Wishlist σου είναι άδεια";
        document.getElementById("wishlist").appendChild(div);
    }
}
```

9.10 Routing

Και τέλος όλες αυτές οι σελίδες καλούνται μέσω του routing σε μία κεντρική σελίδα και σαν αποτέλεσμα έχουμε ενα SPA.

```
import './App.css';
import Thetaclub from './Theta-club';
import OfferCategories from './Offers/OfferCategories'
import OnboardingPage1 from './OnBoardingPages/OnboardingPage1'
import OnboardingPage2 from './OnBoardingPages/OnboardingPage2'
import OnboardingPage3 from './OnBoardingPages/OnboardingPage3'
import SignUp from './SignInPages/SignUp'
import Homepage from './Homepages/Homepage'
import SignUpComplete from './SignInPages/SignUpComplete'
import SignIn from './SignInPages/SignIn'
import SignInSignUp from './SignInPages/SignInSignUp';
import ForgotPass from './SignInPages/ForgotPass';
import Offer1 from './Offers/Offer1';
import Cart from './Homepages/Cart'
import Buy from './Homepages/Buy'
import Tabs from './Homepages/Tabs'
import Carousel from './Homepages/Carousel'
import Wishbuy from './Homepages/Wish-Buy'
import Welcome from './Homepages/Welcome'
import Thetastories from './Homepages/Thetastories';
import Thetanews from './Homepages/Thetanews';
import GalleryDetails from './Homepages/GalleryDetails';
import MyProfile from './Homepages/MyProfile'
import Coupon from './Homepages/Coupon'
import Terms from './Homepages/Terms'
import About from './Homepages/About'
import BuyComplete from './Homepages/BuyComplete'
import BuyCompleteFree from './Homepages/BuyCompleteFree'
import StoryUpload from './Homepages/StoryUpload'
import AfterUpload from './Homepages/AfterUpload'
import Art from './Offers/Art'
import Hashtags from './Offers/Hashtags'

//libraries
import React, { Component } from 'react';
import { AnimatePresence, motion } from 'framer-motion';
import { BrowserRouter as Router, Switch, Route, NavLink, Link, Navbar } from 'react-router-dom'
```

```
return (

    <AnimatePresence exitBeforeEnter>
    <div className="App">
      <Route path="/" exact strict render = {
          () => {
            return(
              <Thetaclub></Thetaclub>
            );
          }
        }/>
        <Route path="/OnboardingPage1" exact strict render = {
          () => {
            return(
              <OnboardingPage1></OnboardingPage1>
            );
          }
        }/>
      <Route path="/OnboardingPage2" exact strict render = {
          () => {
            return(
              <OnboardingPage2></OnboardingPage2>
            );
          }
        }/>
        <Route path="/OnboardingPage3" exact strict render = {
          () => {
            return(
              <OnboardingPage3></OnboardingPage3>
            );
          }
        }/>
        <Route path="/SignInSignUp" exact strict render = {
          () => {
            return(
              <SignInSignUp></SignInSignUp>
            );
          }
        }/>
```

```
<Route path="/Cart" exact strict render = {
  () => {
    return(
      <Homepage>
        <Cart></Cart>
      </Homepage>
    );
  }
}/>
<Route path="/BuyComplete" exact strict render = {
  () => {
    return(
      <Homepage>
        <BuyComplete></BuyComplete>
      </Homepage>
    );
  }
}/>
<Route path="/BuyCompleteFree" exact strict render = {
  () => {
    return(
      <Homepage>
        <BuyCompleteFree></BuyCompleteFree>
      </Homepage>
    );
  }
}/>
<Route path="/Wishbuy" exact strict render = {
  () => {
    return(
      <Homepage>
        <Wishbuy></Wishbuy>
      </Homepage>
    );
  }
}/>
<Route path="/Tabs" exact strict render = {
  () => {
    return(
      <Tabs></Tabs>
    );
  }
}/>
```

```
<Switch>
    <Route path="/Offer1" component={Offer1} />
    <Route path="/Thetastories" component={Thetastories} />
    <Route path="/Thetanews" component={Thetanews} />
    <Route path="/GalleryDetails" component={GalleryDetails} />
    <Route path="/Cart" component={Cart} />
    <Route path="/Buy" component={Buy} />
    <Route path="/Wishbuy" component={Wishbuy} />
    <Route path="/MyProfile" component={MyProfile} />
    <Route path="/StoryUpload" component={StoryUpload} />
    <Route path="/Terms" component={Terms} />
    <Route path="/About" component={About} />
    <Route path="/Art" component={Art} />
    <Route path="/Hashtags" component={Hashtags} />
    <Route path="/BuyComplete" component={BuyComplete} />
    <Route path="/BuyCompleteFree" component={BuyCompleteFree} />
    <div className="offers">
        <img src={logo} className="theta-logo"></img>
        <Carousel></Carousel>
        <Tabs></Tabs>
    </div>
</Switch>
```

Επίλογος

Οι Single Page Applications γίνονται όλο και πιο διαδεδομένες, με έναν τεράστιο αριθμό δημοφιλών ιστότοπων να τις χρησιμοποιεί. Όμως, οι εφαρμογές μιας σελίδας δεν πρέπει να θεωρούνται ως η απόλυτη λύση που θα χρησιμοποιηθεί για όλες τις εφαρμογές ιστού. Αποτελούν περισσότερο ένα ακόμη βήμα στην εξέλιξη του ιστού και μια προσπάθεια να παρέχουν στον χρήστη μια καλύτερη και πιο ρευστή εμπειρία. Το Facebook, το Twitter και το Instagram είναι όλες εφαρμογές μιας σελίδας και η Google τις χρησιμοποιεί επίσης για τις δικές τους υπηρεσίες, συμπεριλαμβανομένου του Gmail και του Google Drive. Μια εφαρμογή μιας σελίδας είναι γρήγορη για φόρτωση και χρήση και δεν ανανεώνεται όταν κάνετε κλικ σε διαφορετικές επιλογές και σελίδες μειώνοντας έτσι τον χρόνο που απαιτείται για τη φόρτωση και το ποσό του εύρους ζώνης.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] Flanagan, David, “JavaScript – The Definitive Guide”, 5th ed. O'Reilly, Sebastopol, CA, 2006, p.497

[2] “Fixing the Back Button: SPA Behavior using Location Hash”. *Falafel Software Blog*.

[3] “Inner-Browsing: Extending Web Browsing the Navigation Paradigm”. Developer.mozila.org

[4] “Slashdotslash.com: A self contained website using DHTML”.

[5] “U.S. patent 8,136,109”. patents.google.gom

[6] “Meteor Blaze”. meteor.com

[7] “Introducing DDP”. meteor.com

[8] “server-side rendering”. meteorhacks.com

[9] “single-page applications vs multiple-page applications”. blak-it.com

[10] “using InnerHTML”. webrocketx.com

[11] “App Store Year Zero: Unsweet web apps and unsigned code drove iphone to an SDK”. imore.gr

[12] “Jobs original vision for the iPhone: No third party native apps”. 9To5mac.com

[13] Marcote, Ethan. “Responsive Web Design”. alistapart.com

[14] Hoffman, Chris. PCWorld | (2016-09-28). “Mozilla is stopping all commercial development on Firefox OS” pcworld.com

[15] “KaiOS, a feature phone platform built on the ashes of Firefox OS, adds Facebook, Twitter and Google apps.” techcrunch.com

[16] Russel, Alex. “Progressive Web Apps: Escaping Tabs Without Losing Our Soul”. infrequently.org

[17] Evans, Jonny. “Apple goes back to the future with web apps” computerworld.com

[18] Ladage, Aaron. “Progressive Web Apps Are Here and They’re Changing Everything”. degdigital.com

[19] “Can I Use...Support tables for HTML5, CSS3, etc.” caniuse.com

[20] Osmani, Addi. “The App Shell Model”. developers.google.com

[21] Russel, Alex. “What, Exactly, Makes a Progressive Web App”. infrequently.org

[22] “Service Worker caching and HTTP caching”. web.dev

[23] w3c. “Web App Manifest”. w3.org

[24] “Discoverable”. developer.mozilla.org

[25] “Web Manifest Docs”. developer.mozilla.org

[26] “Introduction to Service Worker”. developers.google.com

[27] “React – A JavaScript library for building user interfaces” reactjs.org

[28] Krill, Paul. “React: Making faster, smoother UIs for data-driven Web apps”. infoworld.com

[29] Hemel, Zef. “Facebook's React JavaScript User Interfaces Library Receives Mixed Reviews”. infoq.com

[30] Dawson, Chris. “JavaScript’s History and How it Led To ReactJS”. thenewstack.io

[31] Dere, Mohan. “How to intergate create-react-app with all the libraries you need to make a great app”. freecodecamp.org

[32] Walke, Jordan. “FaxJS”. github.com

[33] Papp, Andrea. “The History of React.js on a Timeline”. blog.risingstack.com

[34] Frederic, Lardinois “Facebook announces React Fiber, a rewrite of its React library”. techcrunch.com

[35] “React v16.0”. reactjs.org

[36] “React v16.8”. reactjs.org

[37] “Introducing Hooks”. reactjs.org

[38] “React v17.0”. reactjs.org

[39] “Components and Props”. reactjs.org

[40] “Draft: JSX Specification”. facebook.github.com

[41] “Hooks at a Glance”. reactjs.org

[42] “What the heck is React hooks”. soshace.com

[43] “Using the State hook”. reactjs.org

[44] “Using the Effect hook”. reactjs.org

[45] “Hooks API reference”. reactjs.org

[46] “Hello world, <angular/> is here”. misko.hevery.com

[47] “GetAngular”. web.archive.org

[48] “AngularJS: Developer Guide for v1.5.8: Components”. code.angularjs.org

[49] “angular.js”. github.com

[50] “release v.1.7.9”. github.com

[51] “What is AngularJS”. docs.angularjs.org

[52] “AngularJS”. docs.angularjs.org

[53] “5 awesome angularJS features”. net.tutsplus.com

[54] Hevery, Misko. “Databidning in angularjs”. stackoverflow.com

[55] “Meet the team – Vue.js”. vuejs.org

[56] “Between the wires | Evan You”. web.archive.org

[57] “Introduction – Vue.js”. vuejs.org

[58] “Evan is creating Vue.js”. patreon.com

[59] “What is Vue.js”. www.w3schools.com

[60] “Components – Vue.js”. vuejs.org

[61] “Template syntax – Vue.js”. vuejs.org

[62] “Vue 2.0 is here!”. The Vue Point, medium.com

[63] “Routing – Vue.js”. vuejs.org

[64] trends.google.com

[65] 2020.stateofjs.com

[66] insights.stackoverflow.com

[67] “HTML Web Storage API”. w3schools.com

[68] “.NET Core is the future of .NET”. devblogs.microsoft.com

[69] “”Microsoft Open Sources .NET and Mono by Miguel de lcaza”. tirania.org

[70] “.NET Core is Open Source”. devblogs.microsoft.org

[71] “.NET Core 1.0 released, now officially supported by Red Hat”. arstechnica.com

[72] “Microsoft showcases SQL Server, .NET Core on Red Hat Enterprise Linux deliverables”. zdnet.com

[73] “Announcing .NET Core tools 1.0”. devblogs.microsoft.com

[74] paypal.com