**Αρχιτεκτονική Συστήματος**

**Frontend**

Tο frontend αποτελείται:

* Pages
  + κάθε page έχει το δικό του url
* Components
  + To κάθε page αποτελείται από διάφορα components
* Templates
  + Κάθε page έχει structure ανάλογα με το template που χρησιμοποιεί

Για παράδειγμα η αρχική σελίδα με URL: «/listings»

αποτελείτε από τα εξής components

* Το Navigation bar που βρίσκεται στην πάνω πλευρά της σελίδας
* Το Filters bar το οποίο περιέχει όλα τα φίλτρα που μπορεί ο χρήστης να χρησιμοποιήσει
* Τα listings που ανάλογα με το page που βρισκόμαστε και τα φίλτρα διαφέρουν
* Όλα αυτά για να έχουνε το σωστό structure χρησιμοποιούν το template το οποίο εμπεριέχει αυτά τα components (Μέσω των props περνάει πληροφορία για το τι θα περιέχει το κάθε component, έτσι ώστε να δημιουργείτε δυναμικά η σελίδα)

(προσθέτω μια φωτογραφία που θα δείχνει τα σημεία)

Με παρόμοιο τρόπο χρησιμοποιούνται σε όλες της σελίδες τα components και sub-components.

**Ανάλυση Συστήματος**

Components

Filters:

* FilterCategory.js
  + checkbox με όνομα και value επιλογής
* Filters.js
  + όλη η μπάρα των φίλτρων

Listings:

* Listing.js
  + αποτελεί το κάθε «μεμονωμένο» listing
* ListingDetails.js
  + Όταν ο χρήστης ανοίξει το listing σε full page, αυτό το component εμφανίζει όλα τα στοιχεία του συγκεκριμένου listing
* NewListingField.js
  + Δημιουργεί δυναμικά το κάθε filed που πρέπει να περιέχετε στην σελίδα δημιουργίας νέου listing

Signup:

* SignupField.js
  + Δημιουργεί δυναμικά τα διάφορα πεδία που υπάρχουν στην σελίδα εγγραφής, εκτός πεδίων που ο χρήστης πρέπει να επιλέξει μέσω υπαρχόντων δεδομένων από dropdown menu
* SignupSelect.js
  + Δημιουργεί δυναμικά τα πεδία που είναι για την επιλογή δεδομένων από dropdown menus πχ. Customer ή professional account; Jobs; locations κλπ. που υπάρχουν στην σελίδα εγγραφής

Timetable:

* Timetable.js
  + Δημιουργεί δυναμικά το calendar του professional

Statistics:

* AnaliticsChart.js
  + Δημιουργεί δυναμικά γραφικές παραστάσεις με διάφορα δεδομένα από τον professional πχ πόσα bids έκανε σε listings τον τελευταίο χρόνο

Bids:

* Bid.js
  + Δημιουργεί το κάθε πεδίο που υπάρχει στην φόρμα που συμπληρώνει ο professional
* BidForm.js
  + Δημιουργεί ολόκληρη την φόρμα
* BidsMade.js
  + Εμφανίζει στον customer τα bids που έγιναν στο listing του και του δίνει την δυνατότητα να επιλέξει κάποιο από αυτά

Navigation Bar:

* NavBar.js
  + Δημιουργεί το navigation bar

Reviews:

* Review.js
  + Εμφανίζει το review που έγινε από ένα user προς έναν άλλο
* ReviewForm.js
  + Δημιουργεί την φόρμα για να κάνει κάποιος user review μετά το τέλος ενός listing

Profile:

* UserDetails.js
  + Εμφανίζει τα στοιχεία του χρήστη

Γενικά components:

* Rating.js
  + Εμφανίζει το rating κάποιου χρήστη
* SuccessAlert.js
  + εμφανίζει ένα μήνυμα επιτυχίας πχ μετά από μια επιτυχής εγγραφή, μετά από την δημιουργία καινούριου listing, μετά από ένα bid ακόμα και μετά από edit ενός bid.

Containers

Pages:

* SigninPage.js
  + Σελίδα για signin
* SignupPage.js
  + Σελίδα για signup
* ListingsPage.js
  + Σελίδα με τα διαθέσιμα listings
* NewListingPage.js
  + Σελίδα για την δημιουργία καινούργιου listing
* FullListingPage.js
  + Παρουσίαση όλων τον πληροφοριών ενός listing
* ReviewsPage.js
  + Σελίδα για την προβολή των reviews που έγιναν και που απομένουν να γίνουν
* BidsPage.js
  + Σελίδα με τα bids τα οποία έγιναν και δυνατότητα επεξεργασίας τους
* UserProfilePage.js
  + Σελίδα με τα στοιχεία ενός χρήστη και τα reviews τα οποία έγιναν προς αυτόν
* TimetablePage.js
  + Σελίδα με το πρόγραμμα του χρήστη το οποίο δημιουργείτε δυναμικά με τα bids του που έχουν επιλεγεί
* AnaliticsPage.js
  + Σελίδα με γραφικές παραστάσεις για στατιστικά του χρήστη, δυνατότητα επιλογής χρονικού διαστήματος

Templates

Templates:

* SimpleTemplatePage.js
  + Περιέχει το navigation bar, χρησιμοποιείτε για το NewListingPage, AnaliticsPage κλπ.
* TemplatePage.js
  + Περιέχει το navigation bar και τα φίλτρα, χρησιμοποιείτε για το Homepage, ReviewsPage, BidsPage κλπ.
* FullTemplatePage.js
  + Ίδιο με το TemplatePage με μόνη διαφορά στο margin μεταξύ του navigation bar και του περιεχομένου της σελίδας, χρησιμοποιείτε στο UserProfilePage

State

Global State – Context API:

* Auth-Context.js
  + Περιέχει σε ένα global state τα στοιχεία και το token του χρήστη που έκανε login (Αρχικοποιείται στο App.js)

Local State:

* Κάθε page και κάθε component που περιέχει κάποια φόρμα κρατάει ένα local state. Για παράδειγμα το ListingPage πρέπει να κράτα πληροφορία όπως:
  + Ένα πίνακα με τα listings που πρέπει να εμφανιστούν
  + Αν πατήθηκε κάποιο listing και ποιο είναι αυτό
  + Τα φίλτρα που πρέπει να περιέχει η σελίδα κλπ.

Routing

Page Routing:

* App.js
  + Προσφέρει το routing μεταξύ των σελίδων, προσδιορίζει για κάθε path πιο από τα pages που υπάρχουν πρέπει να γίνει render.

**BackEnd**

Το Backend υλοποιεί ένα REST API το οποίο υποστηρίζει το πιο κάτω methods:

* GET
  + για να πάρουμε ένα resource ή ένα collection από resources
* POST
  + για δημιουργήσουμε ένα resource
* PUT
  + για να κάνουμε update ένα resource
* DELETE
  + για να διαγράψουμε ένα resource

Αποτελείτε από τα εξής packages:

* ServiceLink (main)
  + Περιέχει όλα τα methods μαζί με τα routes που υποστηρίζονται από το REST API, Κάνει redirect το κάθε request στο αντίστοιχο Api. Όταν πάρει την απάντηση από το Api την επιστρέφει στον client
* Data
  + Περιέχει τις κλάσεις που χρησιμοποιούνται για να κρατάνε temporary δεδομένα από και προς την βάση δεδομένων
* Api
  + Λειτουργεί σαν controller μεταξύ του ServiceLink (main) και DB, παίρνει τα requests που γίνονται redirect από την main, ελέγχει αν είναι valid τα στοιχεία και το token και καλεί με την σειρά του μεθόδους από το package DB. Έτσι ετοιμάζει την απάντηση την οποία θα επιστρέψει στην main
* DB
  + Ετοιμάζει και κάνει τα queries προς την βάση δεδομένων, όταν πάρει την απάντηση με την χρήση του package data αποθηκεύει temporary δεδομένα σε maps ή σε απλά objects και τα επιστρέφει στο Api

Κάθε Package αποτελείτε από της εξής κλάσεις

* ServiceLink (main)
  + ServiceLink.java
* Data
  + User.java (abstract)
  + Professional.java (extends User)
  + Customer.java (extends User)
  + Bid.java
  + Listing.java
  + Review.java
  + TimetableEvent.java
  + Token.java
* Api
  + DataApi.java (interface)
  + UserAPI.java (implements UserAPI)
  + BidAPI.java (implements DataApi)
  + ListingAPI.java (implements ListingAPI)
  + ReviewsAPI.java (implements ReviewsAPI)
  + AuthorizerApi.java
  + JwtAPI.java
  + ServerResponseAPI.java
  + StatisticsAPI.java
  + TimetableAPI.java
* DB
  + DB.java (interface)
  + UserDB.java (implements DB)
  + BidDB.java (implements DB)
  + ListingDB.java (implements DB)
  + ReviewDB.java (implements DB)
  + ConnectionDB.java
  + Statistics.java
  + Timetable.java

Διαγράμματα UML

**Database**

Η Βάση δεδομένων αποτελείτε από 5 βασικές οντότητες

* Professional
* Customer
* Bid
* Listing
* Review

Παρότι η βάση δεδομένων αποτελείτε από μόνο 5 οντότητες, παρέχει και επιπλέον δεδομένα τα οποία βγαίνουν από την επεξεργασία των ήδη υπάρχοντών δεδομένων.

Για παράδειγμα το πρόγραμμα των professionals δημιουργείτε δυναμικά, αφού ξέρουμε ότι αποτελείτε από τα bids που έκανε ο professional και επιλέγηκε. Στο κάθε bid περιέχονται πληροφορίες όπως μέρα και ώρα συνάντησης, κόστος επιδιόρθωσής κλπ.

Παρομοίως γίνεται και με τα στατιστικά των professionals αφού τα εξάγονται από τα δεδομένα που υπάρχουν.

**Οντότητες με τα πεδία τους**