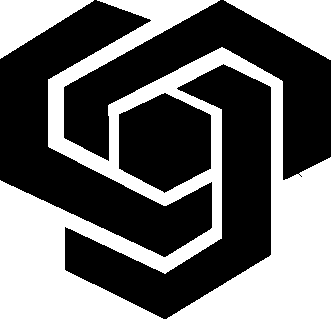
**Технически университет – София**



**Биоинформатика:**

Създаване на уеб портал за подравняване на секвенции

Изготвили:

Николай Гошев Колев фак. № 121210161

Мариян Маринов Маринов фак.№121210074

46 група, КСТ

**София, 2014**

# Въведение

1. **Цел на текущия документ**

Идеята залегнала зад създаването на документацията, която четете в момента, е насочена към запознаване със приложението на всички групи негови потребители, именно програмисти, биолози и медицински лица в сферата на медицинска биология. Конкретните познания, придобити от текущия документ, ще изяснят конкретни въпроси свързани със необходимостта от създаване на приложението, изискванията към него, проучването за съответсващи продукти и/или техни концепции, подход към създаване на необходимите услуги на приложението при наличните ресурси и време, конкретната реализация(използвана технология, интерфейс и вътрешна имплементация), възможности на използване на приложението от различните потребители(user stories/потребителски сценарии). За по-ясно изяснение на концепцията са представени примери и илюстрации към обясненията.

1. **Потребители на документац ията**

* Програмисти разработващи приложението
* Екип поддръжка
* Биолози
* Медицински работници със под-специализация биология

# Дадена задача

12. Информационни портали за Биоинформатика, Портлети, Виртуални среди, Virtual Research Environment.

## Описание

Приложението има две части: потребителска, представена от уеб сайт, достъпен от разстояние през браузър и поддържана на PhP сървър, и back-end част, която извършва същинските изчисления при подадена към сървъра заявка. Интерфейса, който предоставя уеб сайта на потребителите си е базиран на основните настройки, които приложението за изчисление приема. Програмата използвана за подравняване на секвенциите е Clustal Omega, притежаваща конзолен интерфейс и богат избор с допълнитени настройки, които могат да се видят в настоящия портал.

## Желан резултат

От създаденото приложение се изисква:

* Да може да му се подаде на входа секвенция
* Да се зададат и променят настройките за подравняване на секвенцията
* След изпращане да се получи като отговор подредена секвенция

Желателно е, по възможност, също да може:

* Да поддържа голяма опашка със задачи
* Да не използва прекалено много памет и процесорно време на сървъра на които е хостнат сайта
* Да ти връща резултат веднага като стане готов
* Да ползва AJAX при заявките
* Да има разгръщащи полета
* Да има лог-ин за достъп до интерфейса

## Технически изисквания

* Приложението да е достъпно през браузър

# Проектиране и дизайн

## Изследване на налични решения

За текущото изследване ще разгледаме уеб портала Националният център за биотехнологична информация на САЩ (<http://www.ncbi.nlm.nih.gov>), чиито сайт дава достъп до централния институт за обработка и съхранение на данни за молекулярна биология.

Някои от задачите, с които се занимава NCBI са:

* Създаване на автоматизирани системи за съхранение и анализ на данни по молекулярна биология, биомедицина и генетика.
* Компютърна обработка на данни, получени при изследвания на биологически активни молекули и вещества.
* Съдействие за широко използване на базите данни и програмно обезпечаване и подпомагане на изследователи в областта на биотехнологията и медицински работници.
* Координиране събирането на биотехнологична информация по целия свят.

Също NCBI предоставя информация за бази данни за ДНК и РНК, бази данни за статии научна литература и таксономична информация, обезпечава търсене на данни за конкретни биологични видове, а също така съдържа различни програми по биоинформатика. Базите данни са достъпни чрез търсачката Entrez.

## Целеви групи

Целевите групи, към които се стремим са научни изследователи, занимаващи се в областите на генетиката, молекулната биология, медицината и други пряко или непряко свързани с тях области.

## Потребителски групи

Потребителя може да:

* Достъпва сайта
* Качва файлове със секвенции
* Задава настройки на обработката на секвенциите
* Изпраща за обработка дадена секвенция
* Види резултата от заявката

## Ресурси

За ресурси за разработка на приложението имаме ограничено време със две човешки сили за разработване главно на уеб сайта. От гледна точка на приложението на програмата е силно намалено като ресурс времето необходимо за създаване на индивидуален дизайн чрез преизползване на готов дизайн и адаптирането му към нуждите ни. Ограничения от характер на технически ресурс и опит премахват и варианта за по-пълни реализации като паралелна обработка на заявките, отделяне на хостинг машината от машината за изчисление на резултата, интегриране на други инструменти и функционалности, възможност за смяна на езика, и др.

## Текущи цели- изисквания

Считайки целта на проекта, неговото приложение и средства, за текущата версия на продукта е предвидено да се реализира:

* Интерфейс предоставящ повечето от настройките осигурени от конзолния интерфейс на ClustalO
* Възможност за качване на файл със дадените секвенции
* Получаване на резултат със Ajax заявка

## Инсталация на сайта

За инсталация под Windows:  
1. Инсталирайте WAMP: <http://www.wampserver.com/en/>  
2. Стартирайте WAMP server (В Windows менюто -> start WampServer)  
3. Преместете съдържанието на репо-то на проекта (https://github.com/merx3/clustalo) в C:\wamp\www\clustalo (създайте папката clustalo)  
4. Отворете браузъра и отидете на <http://localhost/clustalo>

## Кратко описание на продукта

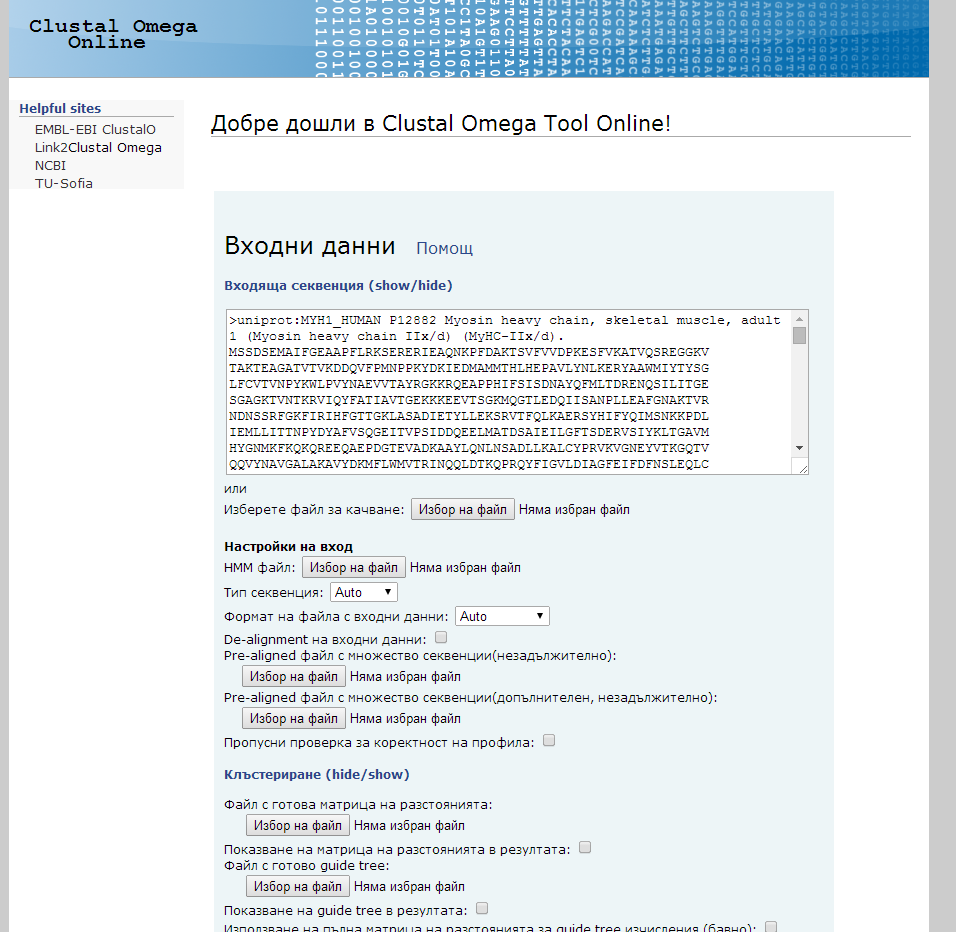
Продукта е достъпен за потребители на <domain-name>/clustalo. Самия продукт използва Clustal Omega (<http://www.clustal.org/omega/>) приложението, което има богат конзолен интерфейс за подаване на задачи за подравняване на секвенции (<http://www.clustal.org/omega/README>). Уеб портала, който се генерира от PhP сървър, предоставя повечето настройки на една стандартна конзолна команда в Clustal Omega. С възможност за качване на секвенцията, Pre-aligned файл с множество секвенции, файл с готова матрица на разстоянията и файл с готово guide tree. Потребителя може да прати своя рекуест, който се обработва с приложението, като ползва Ajax при получаване на резултата. Резултата се представя в текстов прозорец на същата страница със готовата подредена секвенция. Приложението също има и линк за Помощ с по-детайлни описания на настройките.

## Бъдещи възможни подобрения

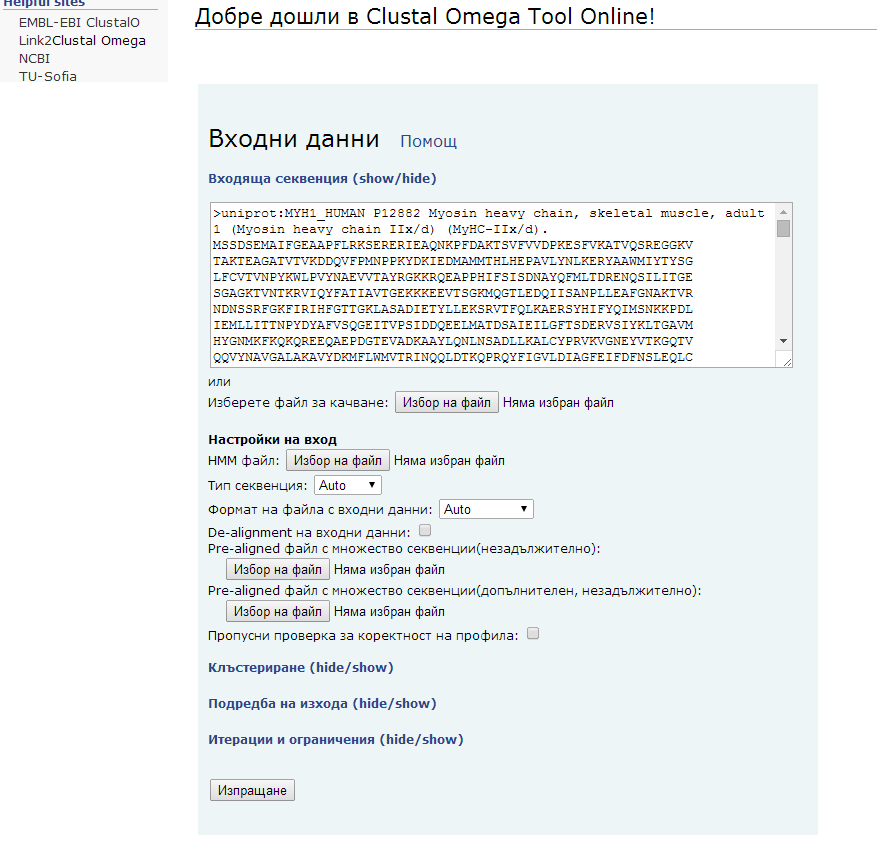
Възможни подобрения по ститемата, които не са включени в текущото приложение е:

* Създаване на сесия помнеща настройките на юзъра
* Да поддържа голяма опашка със задачи
* Да праща на потребителя мейл за готова задача, без Аjax
* Да има лог-ин за достъп до интерфейса
* Резултатите да се пазят в архив заедно със заявките

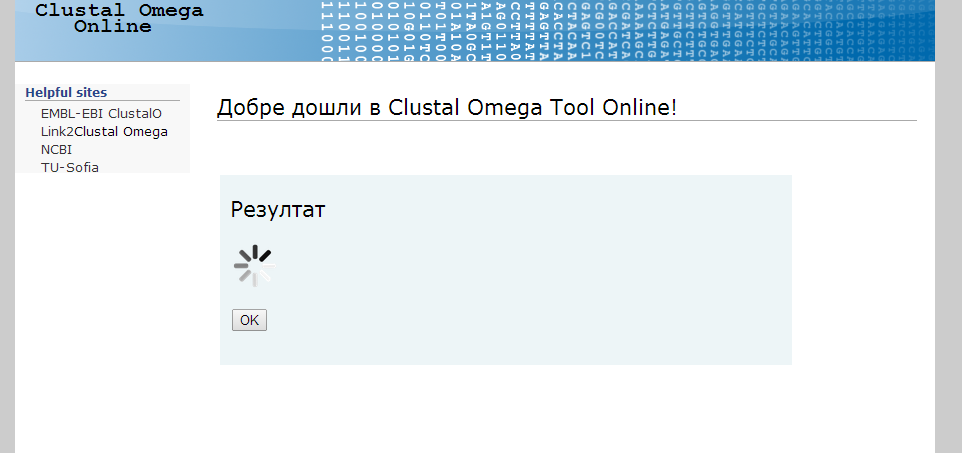
# Изглед и употреба



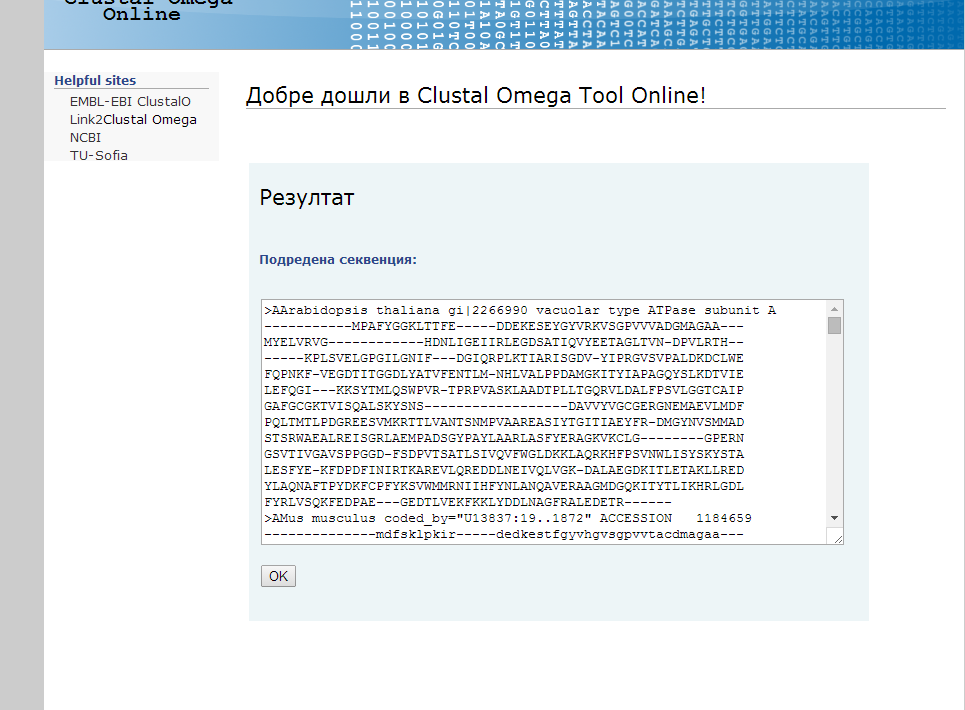
Фиг. 1 Началната страница с текстово поле за вход, както и допълнителни настройки (информация за настройките може да намерите на <http://www.clustal.org/omega/README>)



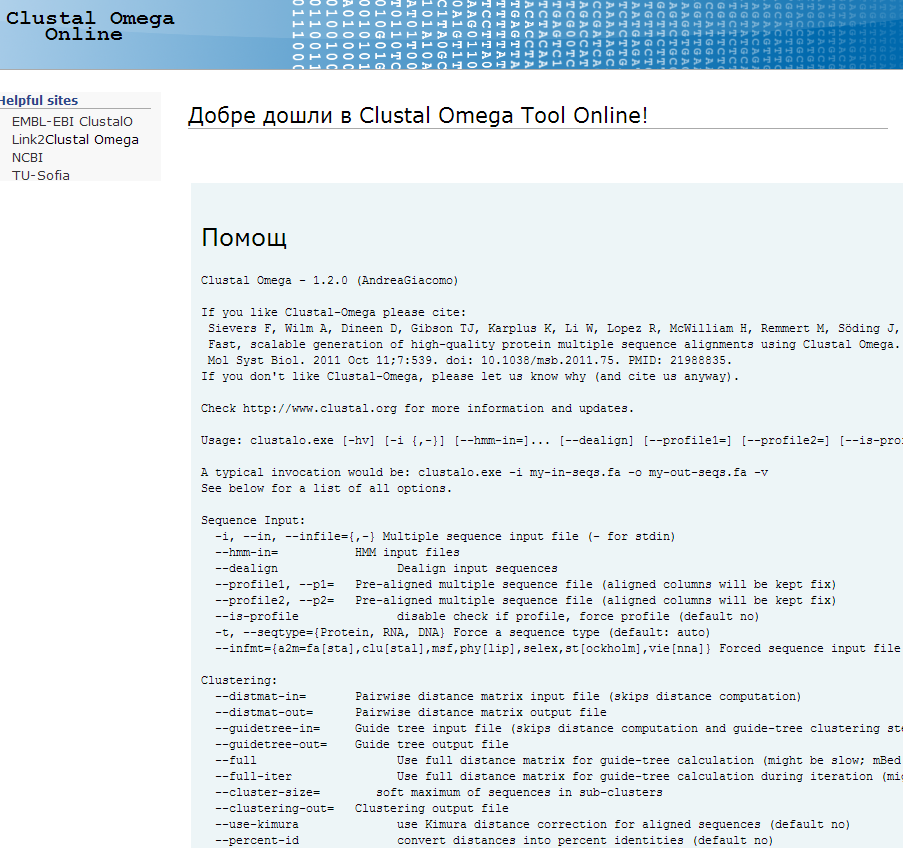
Фиг. 2 Компактен вид, постигнат с натискане на бутоните (hide/show)



**Фиг.3 Обработка на изпратен резултат**

****

**Фиг.4 Получаване на краен резултат**

****

**Фиг.4 Меню за помощ**

# Програмна реализация

Стуктурата на портала има back-end част(програма) и web хост част. Основната страница се съставя от:

* home.php – станицата без каквото и да е съдържание
* content.php – съдържанието което се попълва при първоначалното отваряне на home.php
* result.php – съдържанието което се показва след получаване на резултата
* help.php – показва в съдържанието помощен текст
* calculate.php – съдържа логиката за обработка на рекуеста, сглобяването и подаване на командата, получаване и обработка на резултата за показване в result.php

## Методи

function toggleMe(a){

var e=document.getElementById(a);

if(!e)return true;

if(e.style.display=="none"){

e.style.display="block";

}

else{

e.style.display="none";

}

return true;

}

Използва се за промяна на видимостта на някои елементи на страницата

function prepareAjax(){

// variable to hold ajax request

var request;

// bind to the submit event of our form

$("#input").submit(function(event){

// abort any pending request

if (request) {

request.abort();

}

// setup some local variables

var $form = $(this);

// let's select and cache all the fields

var $inputs = $form.find("input, select, button, textarea");

// serialize the data in the form

var serializedData = $form.serialize();

// let's disable the inputs for the duration of the ajax request

// Note: we disable elements AFTER the form data has been serialized.

// Disabled form elements will not be serialized.

$inputs.prop("disabled", true);

// prevent default posting of form

event.preventDefault();

var file\_input\_names = new Array("upl\_in", "pre\_align\_in1", "pre\_align\_in2", "dist\_matr\_in", "hmm\_file\_in", "guide\_tree\_in");

read\_all\_files(file\_input\_names);

function read\_all\_files(inputs){

if(inputs.length == 0){

continue\_ajax\_after\_file\_reading();

return;

}

var input\_name = inputs.pop();

var files = $("[name='" + input\_name +"']")[0].files;

if(files.length > 0){

var f = files[0];

if (f) {

var reader = new FileReader();

reader.onloadend = function(evt) {

// file is loaded

result = evt.target.result;

var content = encodeURIComponent(result);

serializedData += ("&" + input\_name + "=" + content);

read\_all\_files(inputs);

};

console.log("reading!");

reader.readAsText(f);

} else {

alert("Error! Could not read file.");

}

}

else{

// file input was not uploaded(has no file specified), continue reading other file inputs

read\_all\_files(inputs);

}

}

function continue\_ajax\_after\_file\_reading(){

console.log(serializedData);

// fire off the request to /form.php

request = $.ajax({

beforeSend: function(){

$('#result').show();

$('#wait\_div').show();

$("#input\_wrapper").hide();

},

url: "/clustalo/web/includes/calculate.php",

type: "post",

data: serializedData

});

// callback handler that will be called on success

request.done(function (response, textStatus, jqXHR){

$('#result\_output').html(response);

$('#wait\_div').hide();

});

// callback handler that will be called on failure

request.fail(function (jqXHR, textStatus, errorThrown){

// log the error to the console

console.error(

"The following error occured: "+

textStatus, errorThrown

);

});

// callback handler that will be called regardless

// if the request failed or succeeded

request.always(function () {

// reenable the inputs

$inputs.prop("disabled", false);

});

}

});

}

Функция за подготвяне на Ajax заявка, която също включва и паралелно четене и добавяне на файла в POST рекуеста. Понеже четенето на файл е асинхронен процес, пращането на рекуеста е разцепено в няколко event handlers.