



WEST UNIVERSITY OF TIMIȘOARA
FACULTY OF MATHEMATICS AND COMPUTER
SCIENCE
BACHELOR/MASTER STUDY PROGRAM: Computer
Science in English / Artificial Intelligence and Distributed
Computing / Big Data - Data Science, Analytics and
Technologies

BACHELOR THESIS

SUPERVISOR:
Prof./Conf./Lect. Dr. Mihail Gaianu

GRADUATE:
Abdelhak Haddadi

TIMIȘOARA
2020

WEST UNIVERSITY OF TIMIȘOARA
FACULTY OF MATHEMATICS AND COMPUTER
SCIENCE
BACHELOR/MASTER STUDY PROGRAM: Computer
Science in English / Artificial Intelligence and Distributed
Computing / Big Data - Data Science, Analytics and
Technologies

Title

**chat robot of scientific translation
website**

SUPERVISOR:
Lect. Dr. Mihail GAIANU

GRADUATE:
Abdelhak Haddadi

TIMIȘOARA
2020

Contents

List of Figures	3
List of Tables	3
1 Abstract	6
1.1 the problem	6
1.2 Thesis Scope and Objectives	6
2 Website content	7
2.1 Home page	7
2.1.1 news	7
2.1.2 works	8
2.2 services	9
2.3 About us	10
2.4 Sign up	11
2.5 Sign in	12
3 Relevant Diagrams	13
3.1 activity diagram	13
3.2 sequence diagram	14
4 App Requirements	15
4.1 Back end	15
4.2 front end	16
4.3 database	16
5 application architecture	17
5.1 Use Cases Diagram	18
6 Preliminary App Architecture	19
6.1 Presentation Layer	19
6.2 Application Layer	19
6.3 Data Layer	19

List of Figures

2.1	news picture	7
2.2	works picture	8
2.3	services picture	9
2.4	about us picture	10
2.5	sign up picture	11
2.6	sign in picture	12
3.1	business processes diagram	13
3.2	sequence diagram	14
5.1	application architecture	17
5.2	use case diagram	18

List of Tables

2.1 table of services	10
4.1 table of dependencies	15
4.2 table of libraries	16
4.3 table of libraries	16

Chapter 1

Abstract

1.1 the problem

A team of students that provides in biology major various services to university students such as scientific translation, statistics, methodology, bio-informatics. the company is having some difficulties of communicating with customers and meeting their needs. They have come up with the idea of communicating through social media platforms, but it is not effective. They may need a platform in which all services can be displayed, and the customer can upload files and specify his needs without the need for a direct conversation with team members.

1.2 Thesis Scope and Objectives

The idea of the project is a build a website application for this team so that they can display their work, services, team members.the client can create an account select his needs via robot service chat, upload his document and send to the team and also the admin check the requirements of the client and sends a response(accept or reject),if the team accept the request they will start working on, there is a possibility of chatting with the client during the progressing.

Chapter 2

Website content

2.1 Home page

It is divided into two parts

2.1.1 news

it's basically a collection of photos changes automatically where the user can see all the news,the updates on this part of the page, and the admin can update (add or remove) pictures. when the client click on home button the component of home has typescript class that is connected with beck-end sends a get request to java spring and java fetch data from SQL server . I used "carousel slide" slideshow to display the pictures.

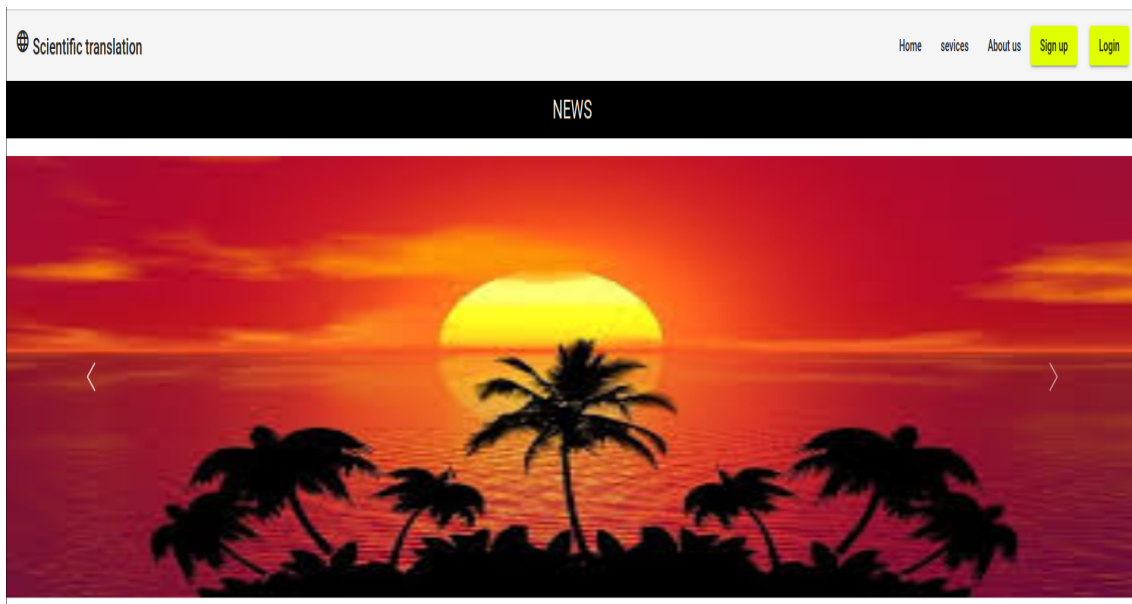


Figure 2.1: news picture

2.1.2 works

in this part of page there are some of the company achievements, it is a collection of cards under each other the picture on the left side and the title and description on right side the admen can update (add or delete) the cards.



Figure 2.2: works picture

2.2 services

It is a new page, it is a collection of cards next to each other, each card has picture of the service on the top and a title and the details of the service in the bottom of the page there is a robot chat service floating button where the client after login can choose the service and the requirement upload the document and send it to the team by answer on the questions of the robot. the team has a different page they have a collection of cards next to each other on top there are the details of client under the client info there are the details of the and the documents that the team download it.

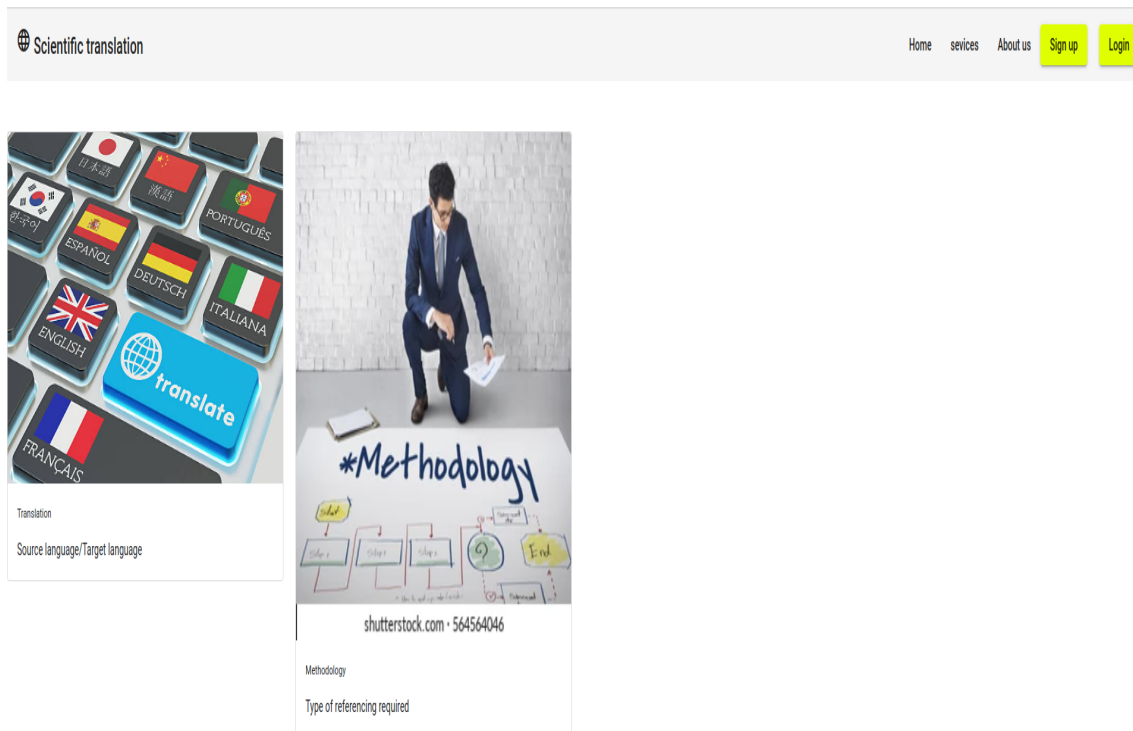


Figure 2.3: services picture

services	description
Translation	Source language/Target language
Methodology	Type of referencing required
Statistics	Analysis required
Bioinformatics	DNA sequencing

Table 2.1: table of services

2.3 About us

this page had been divided to parts, the first part is an introduction to the team, the second part contains the team members, it is a collection of cards so that the image is from the left side and the name details from the right side when we minimise the browser the image will be on the top the name and the details will be on the bottom of the cards, each cards has an action listener can direct the user to chat with the team member.

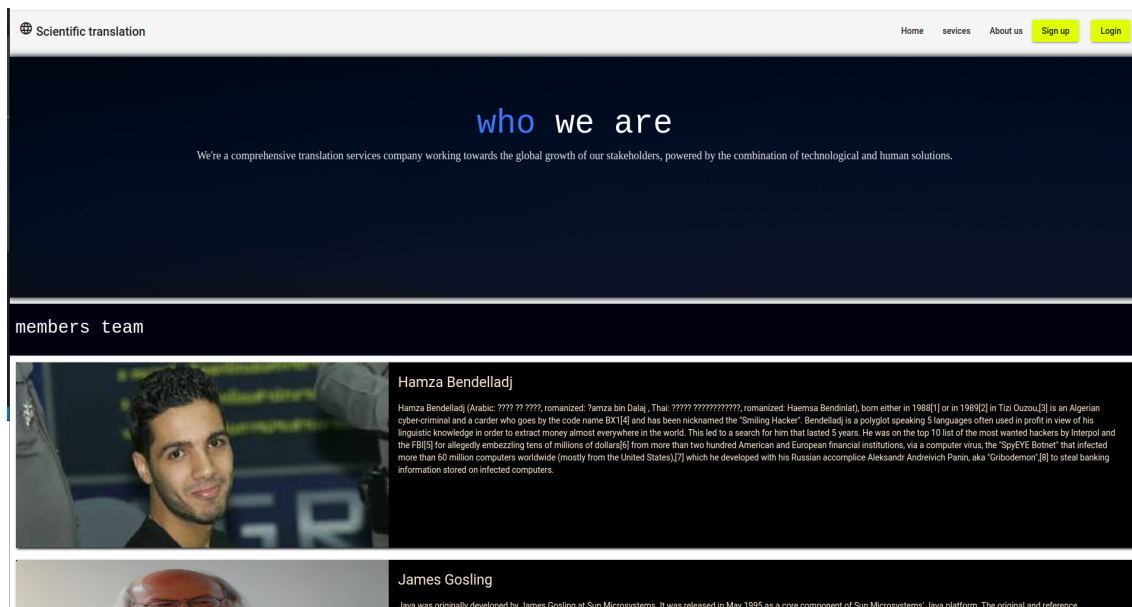
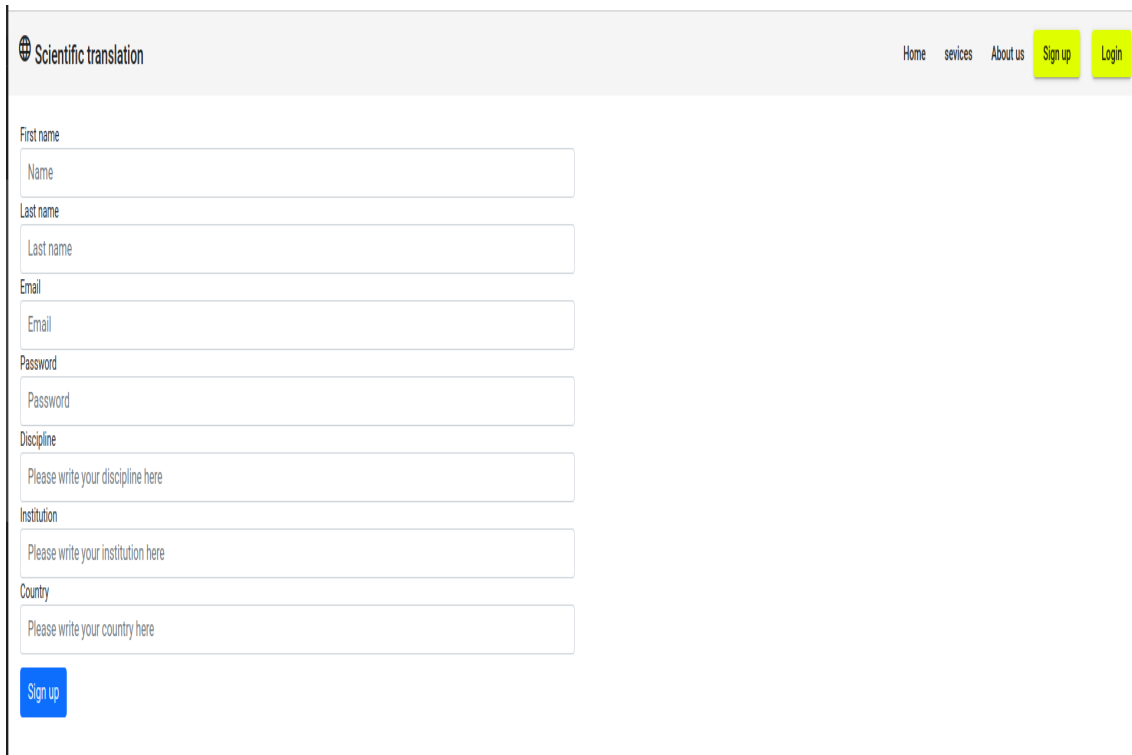


Figure 2.4: about us picture

2.4 Sign up

the client can create an account on this page, where all the fields are required if the client didn't filled one of them and he clicked on submit button the page will display an error message under the field that he didn't filled, when the submit button is clicked the front-end (angular) will send post request to java spring, the back-end will encrypt the password using secret key before save it in the database if the registration has been passed successfully the page will be reloaded.

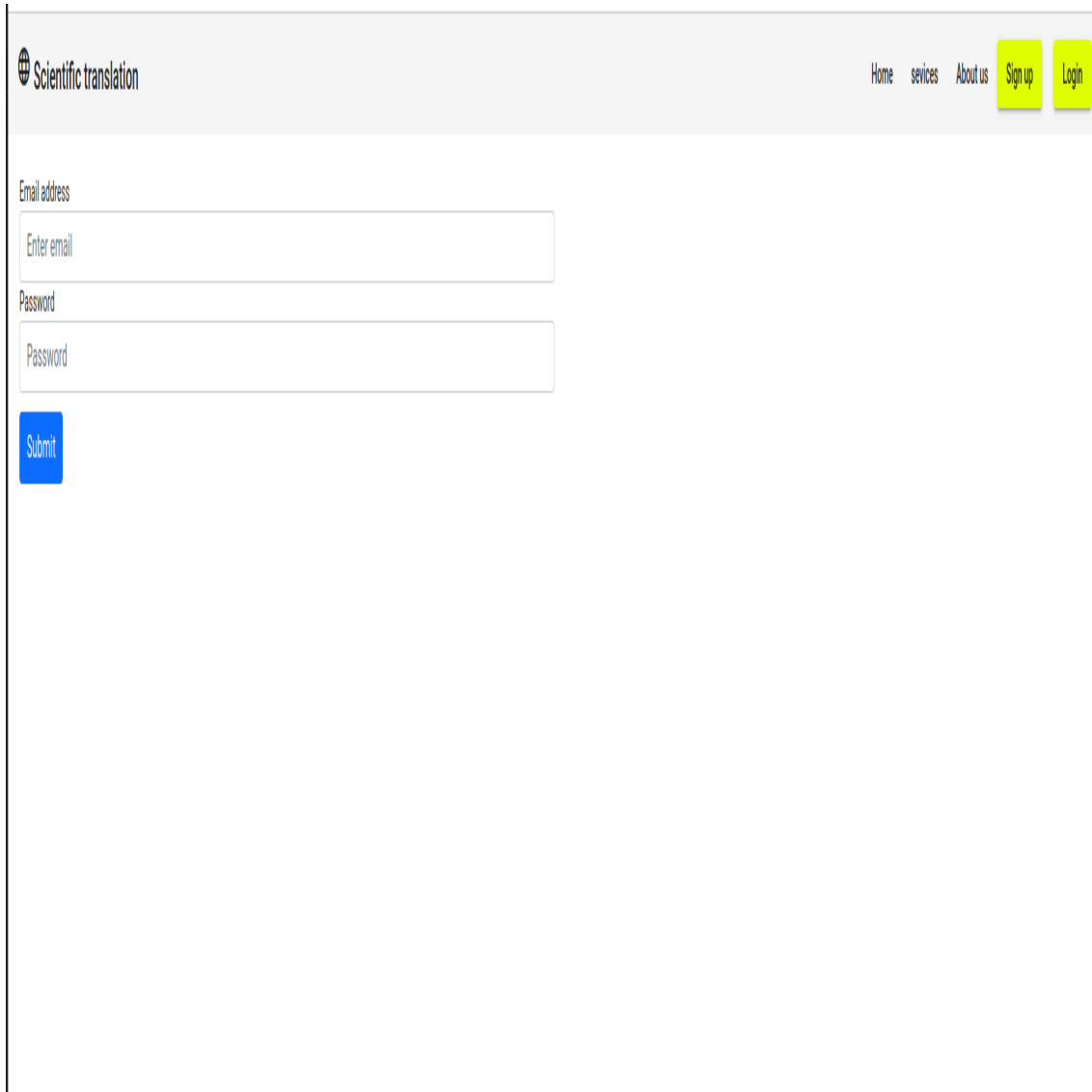


The screenshot shows a web page for 'Scientific translation'. The header includes a logo and navigation links: Home, services, About us, Sign up, and Login. The main content area contains a sign-up form with the following fields: First name (placeholder: Name), Last name (placeholder: Last name), Email (placeholder: Email), Password (placeholder: Password), Discipline (placeholder: Please write your discipline here), Institution (placeholder: Please write your institution here), and Country (placeholder: Please write your country here). A blue 'Sign up' button is located at the bottom left of the form.

Figure 2.5: sign up picture

2.5 Sign in

In this page the client should fill email and password fields when the login button is clicked angular will send a post request to java spring(spring security), it will decrypt the password that it has read from database using secret key that it used for registration.



The image shows a web form for signing in. At the top, there is a header bar with the logo 'Scientific translation' on the left and navigation links 'Home', 'services', 'About us', 'Sign up', and 'Login' on the right. The 'Sign up' and 'Login' buttons are highlighted in yellow. Below the header, the form consists of two input fields: 'Email address' and 'Password'. The 'Email address' field has a placeholder text 'Enter email'. Below the 'Password' field is a blue 'Submit' button.

Figure 2.6: sign in picture

Chapter 3

Relevant Diagrams

3.1 activity diagram

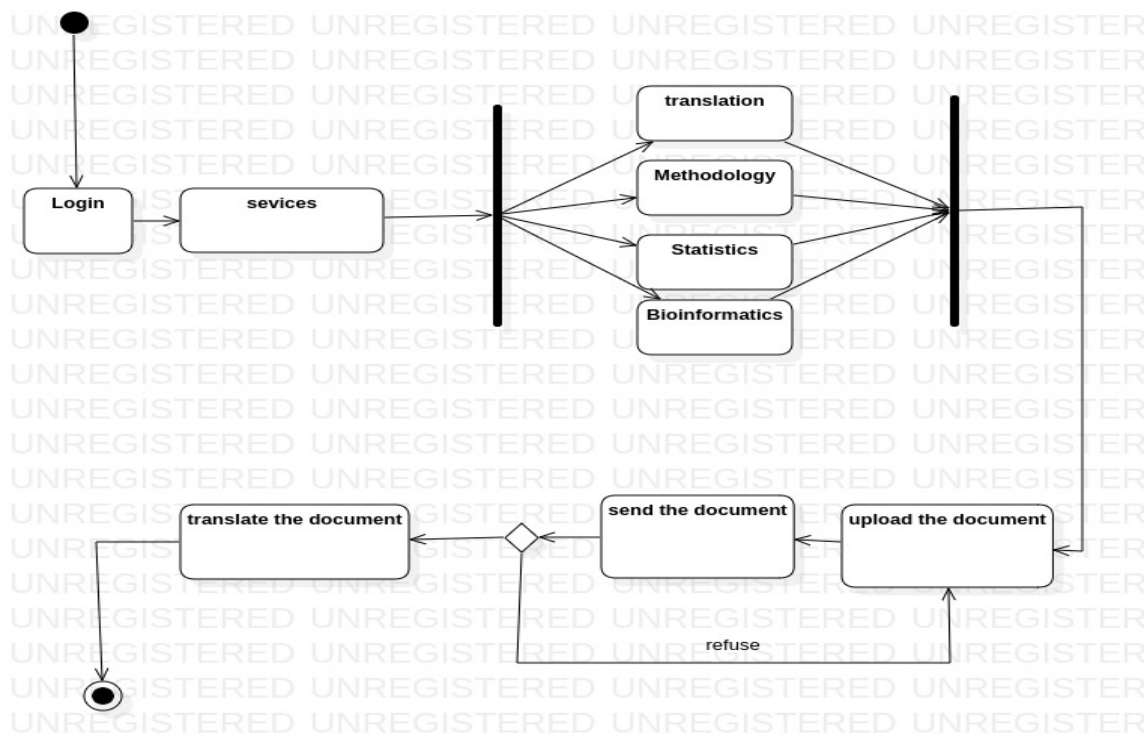


Figure 3.1: business processes diagram

in the the starting point the user should open the browser and the first activity is login after login he should to services page and in that page there is fork node where the client can select the services and the requirements, after selecting there is join node, the next activity is upload the document and send the document activity when the admin receive the document there is decision he can accept it or reject it, if accept it go to the next activity translate the document or reject go back to the upload the document activity.

3.2 sequence diagram

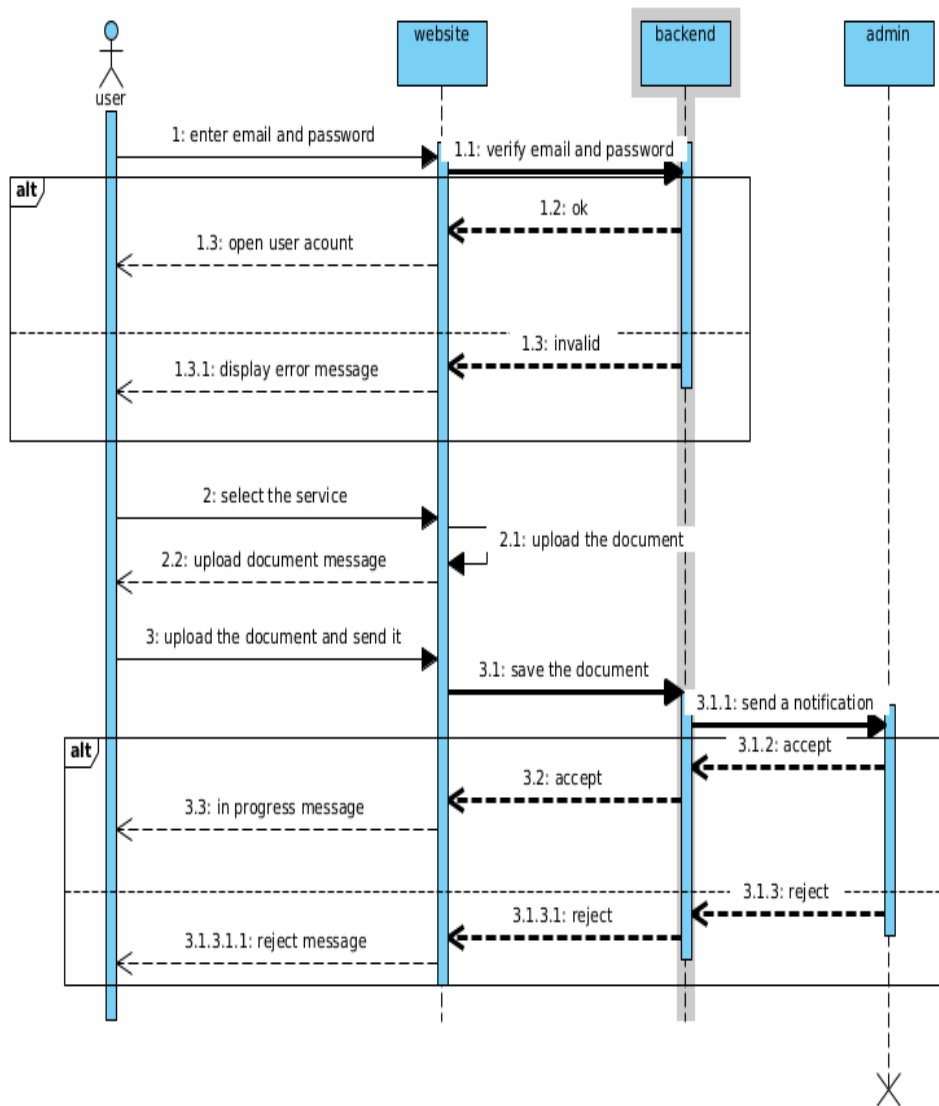


Figure 3.2: sequence diagram

there are two actors and two objects the first actor is the user and website object have an active life line throughout operation, enter the email and password the website will and them to back-end are represented by two arrows , the backend will replay by accept or reject are represented by dashed line arrows and alternative frame, when the user finish selecting the service we have self message to upload the document is represented by rounded arrow, after upload the document and send it to admin, the admin will replay by accept or reject the message is represent by dashed arrow and the decision by alternative frame.

Chapter 4

App Requirements

4.1 Back end

Spring boot: JDk should be installed java 8 and above, IntelliJ IDEA IDE.
dependencies required:

dependency	description
spring-boot-starter-web	Starter for building web, including RESTful, applications using Spring MVC
spring-boot-starter-jdbc	ability to connect with database (sql server)
spring-boot-starter-web-services	is a product of the Spring community focused on creating document-driven Web services
spring-boot-devtools	The spring-boot-devtools module includes an embedded LiveReload server that can be used to trigger a browser refresh when a resource is changed.
spring-boot-starter-test	is the primary starter dependency for testing our spring boot application
lombok	the tool of the java library that was used to generate code for minimizing the unused code

Table 4.1: table of dependencies

4.2 front end

Angular: Node.js should be installed, visual code IDE
libraries required:

dependency	description
NgModule	NgModules consolidate components, directives, and pipes into cohesive blocks of functionality, each focused on a feature area, application business domain, workflow, or common collection of utilities.
BrowserModule	Exports required infrastructure for all Angular apps. Included by default in all Angular apps created with the CLI new command
MatToolbarModule	is a container for headers, titles, or actions. Toolbar overview.
MatButtonModule	Overview for button.
AppComponent	any component that Angular loads imperatively.

Table 4.2: table of libraries

4.3 database

Angular: SQL server should be installed
SQL required tables:

tables	description
card	this table for services cards and it contains image url, title and description as columns
definition	this table for news part, it contains image url and description
homePage	this table for work part, it contains image url, title and description

Table 4.3: table of libraries

Chapter 5

application architecture

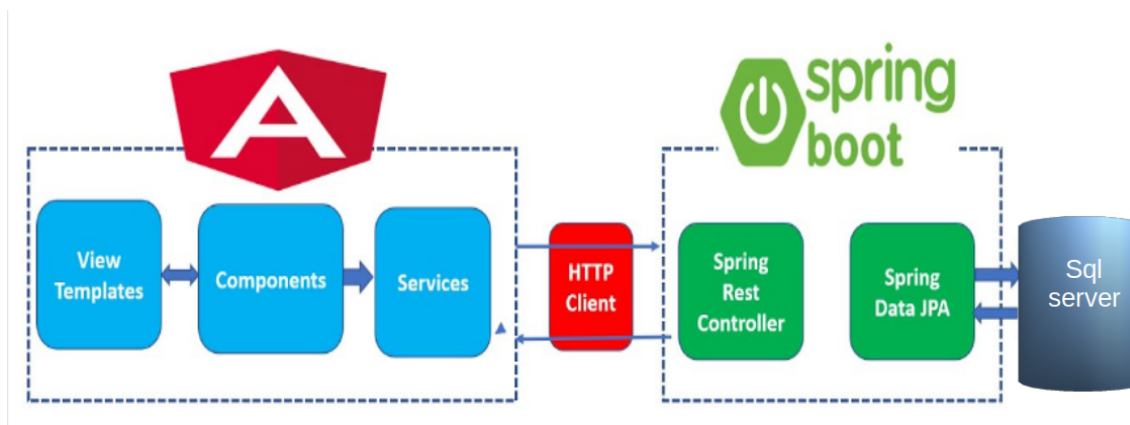


Figure 5.1: application architecture

5.1 Use Cases Diagram

se Diagram Example| /

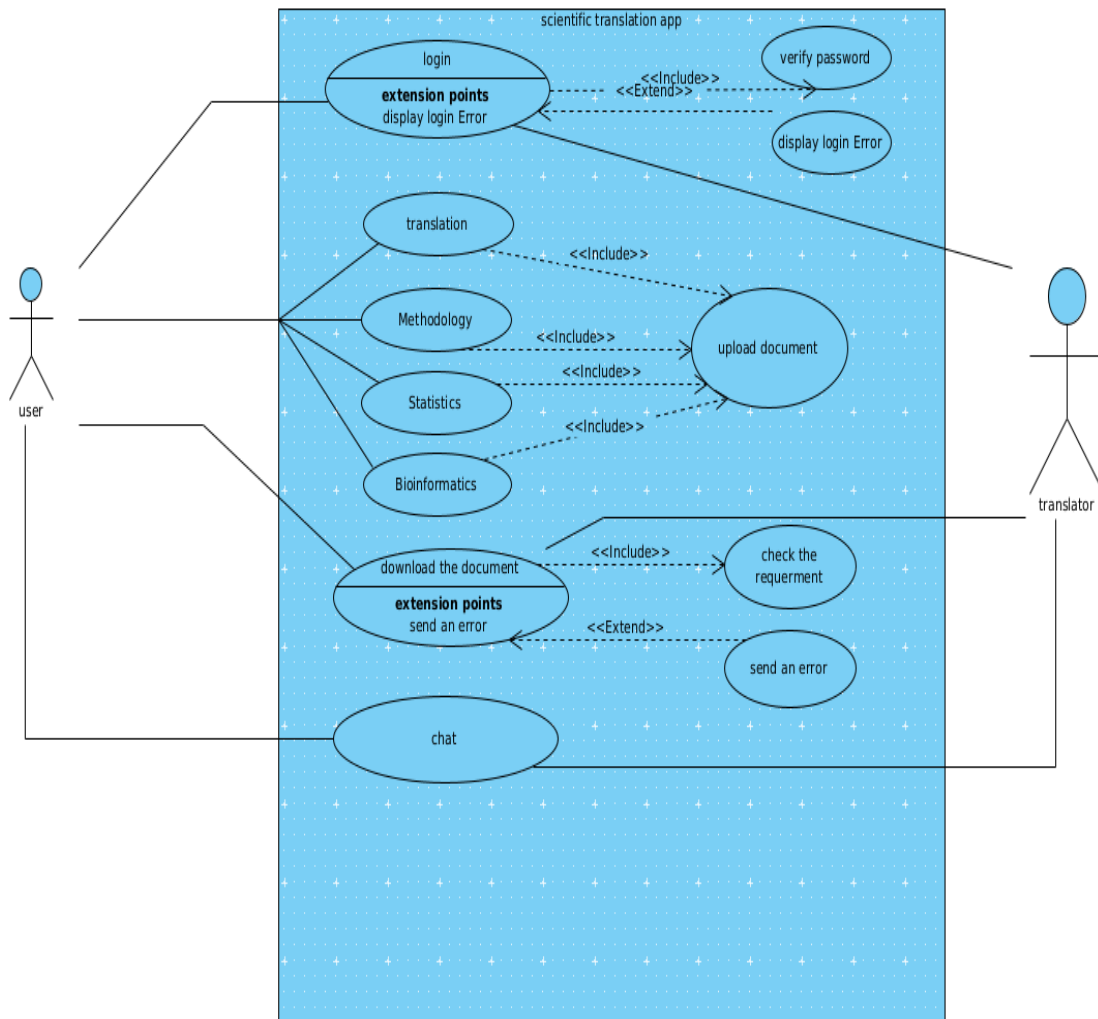


Figure 5.2: use case diagram

Chapter 6

Preliminary App Architecture

6.1 Presentation Layer

The frontend of application(Angular), this layer to display the data it retrieves and do as little "thinking" as possible.

6.2 Application Layer

This is the middle-ware of the application(java spring), This layer is the "engine" of webApp and it connects the presentation layer to the data layer

6.3 Data Layer

database or storage system(SQL server),The data would be accessed by the application layer via API calls (note: the API call itself is triggered by the presentation layer, but the presentation layer doesn't know what the application layer will do with that call, it's a blackbox between the two).