UTALK

# Internet technologies - project

Team :

Amariei Marinel – Team Leader

Blejusca Oana

Ciobanu Denis

We chose to make our own version of a virtual social network to find out more about how these big companies succeeded and the work behind those beautiful html landing pages.

Preexisting such applications are already well known, such as Facebook, Twitter, Instagram. They excel through a big, scalable infrastructure that can sustain big loads of requests and accesses every second.

## Specifications and requirements

Our application is going to be based on:

-database server

-web services server

-web server

-web client(browser)

-mobile app

Main implemented features:

-Personal profile – Register, Create and Edit profile

-Friends list – Add, remove, search friends

-Messaging – Send and post messages

Hardware/software resources:

-Visual Studio Code, Sublime and Notepad++ – text editor

-IntelliJ/Eclipse – Java IDE

-ApacheTomcat 8.5 – server

-MySql – Database server

-Github – Project management

- Google Chrome – Browser

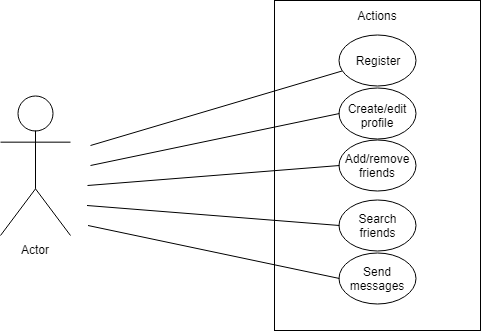
The main protagonist is our user. He has the possibilities to register or log in, in order to have access to his information.

Application components: User interface, Mobile application, Web server, Web services server, Database server.

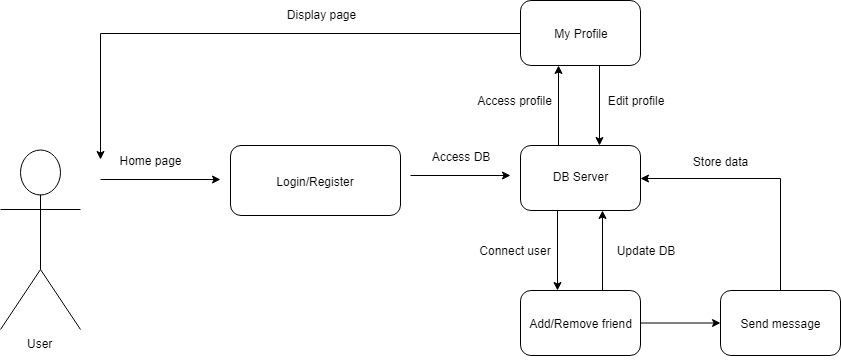
## UML DIAGRAMS

• Use-case diagram: This diagram offers a general description of the system way of usage, provides an overview of the features that the system offers and shows how it interact with the user.

The user has the possibility to create a profile, to publish posts/articles, he can visualize the friends list and add new friends in list. All the operations are available, depending of the each user rights, at the graphic interface level, through hyperlinks, buttons or forms.



• Information-Flow Diagram: Emphasizes the path the information covers from source to destination.



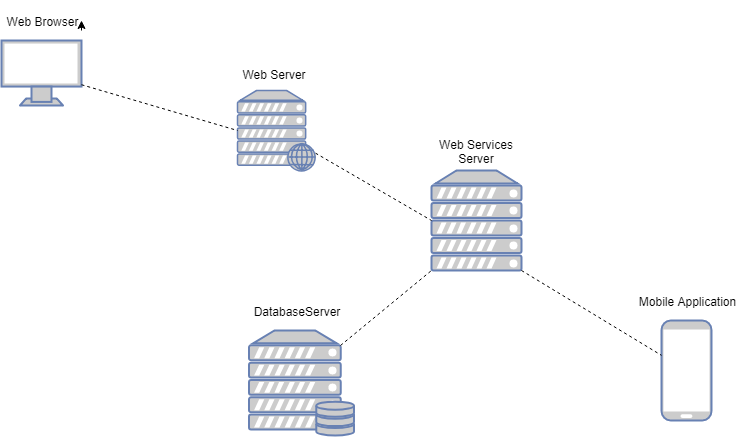
• Components Diagram: It shows the app components and the dependencies between them. A component is a class that represents a modular part of a system with encapsulated content. Utalk has as principal components a web services server, a web server and a database server. A separate component will be the mobile app. There are 5 interconnected components. This diagram has the role to present the physic structure of the code and its grouping per components using a mapping on the logic view of components.

A component can contain a source code or it can be in binary or executable form.

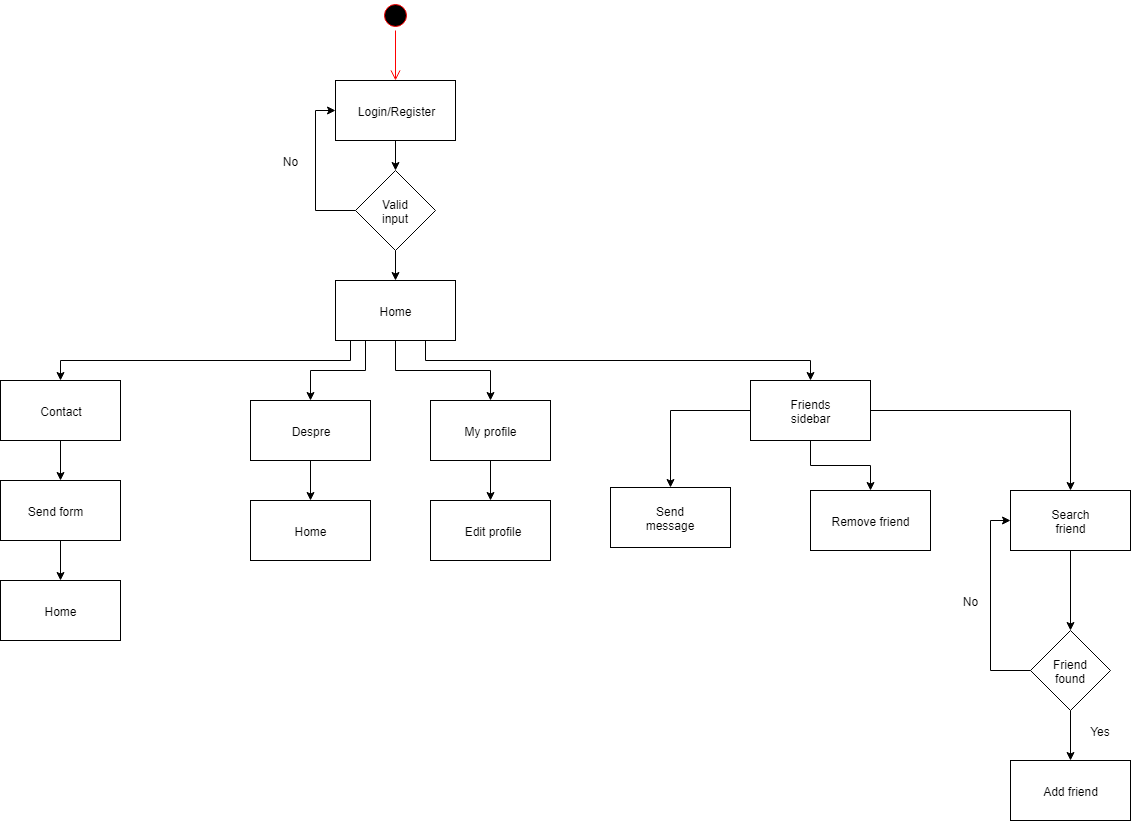
The diagram describe the path of the information, the relationships and dependencies between components.

The user use an interface given by the app or the web server, so he can log in.

Any action of the user is transmitted from the web services server that transmits further to the database server. There the information processing the answer is transmitted to the web services server that will transmit to the user messages that describe the action results.

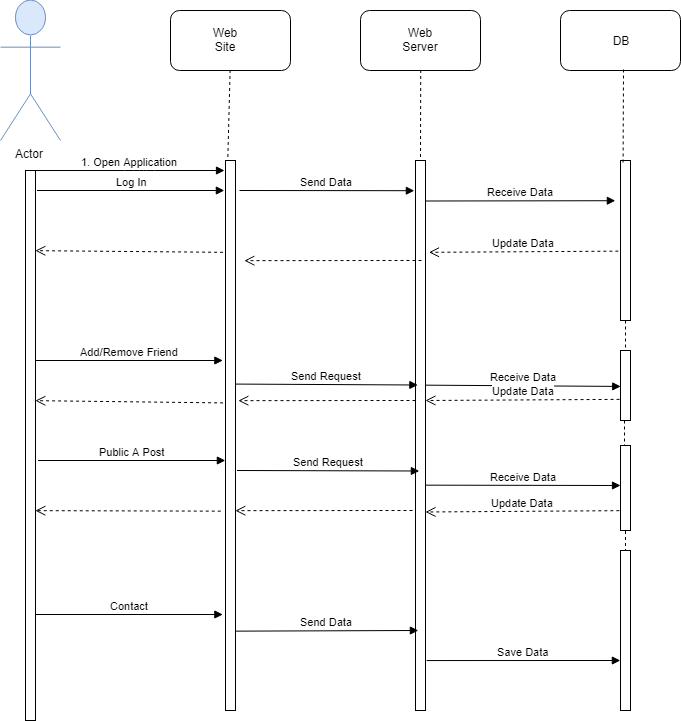


• Activity diagram: It presents the actions that the user can do on the site, together with the possible transactions and the conditioning between them.

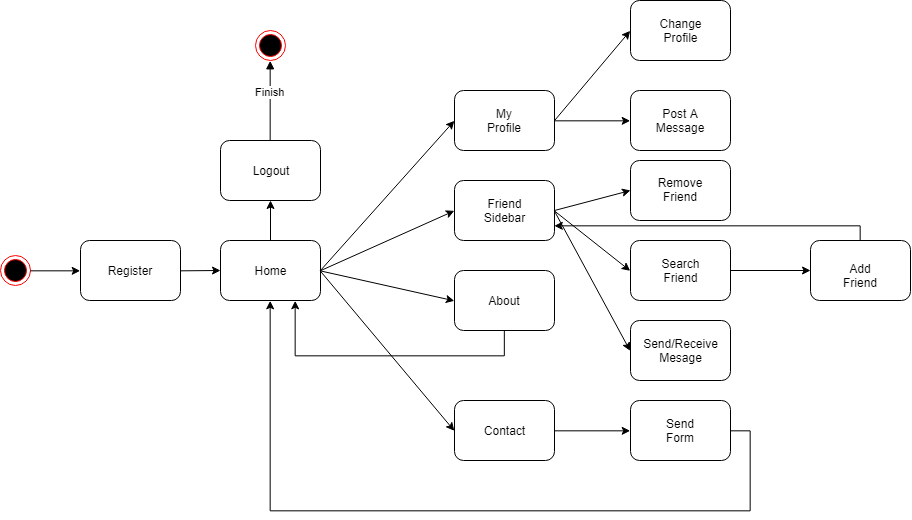


• Sequence diagram: It focus on the messages exchange between classes, components, subsystems or actors. The transmitted messages are organized chronologically from up to down, represented through arrows with continuous line, drawn from source to destination. Messages that represents responses are drawn using interrupted line.

The entities that exchange messages are: user, app ( web or android ), the services server and the entire database.



• State machine diagram: It shows the states sequence that the system can take, from the user point of view. At the same time, the events caused by the user that lead to state change, the final state getting reached at logout.



• Entity-Relationship diagram: This diagram displays the layout of our database which consists of 4 tables: Users (with id, name, password and profile id), Profile(user-id – foreign key to users.id) one to one relationship with Users, Messages (sender-id, receiver-id – foreign key to users.id) many to one with Users, Posts(user-id – foreign key to users.id) many to one with Users.

