

Kyle Connect

Design Document



Course: Bachelor of Science (Honours) in Software Development

Name: Kyle Kinsella

Student Number: C00273146

Date: 14/02/2025

Supervisor: Joseph Kehoe



Introduction	2
Technology Stack: Implementation languages	2
Frontend languages and frameworks	2
HTML: Hyper Text Markup Language	2
CSS: Cascading Style Sheets	3
Backend languages and frameworks	3
Go: Golang	3
Deployment	4
Docker	4
Why am I using Docker?	4
Algorithms	5
Pseudocode	6
Database	7
My-SQL	7
User-interface design	8
Wireframe	8
Welcome page	8
Sign up	8
Login	9
Main page - user interface	9
Send friend request	10
Send message to a friend	10
Create a server	11
Add friends to server	11
Delete friends from server	12
Target platform	13
How Technology Stack communicate together	13
Sign up page	13
Login page	14
Main page - user interface	15
Send Friend Request page	16
Send message to a friend page	17
Create a server page	18
Add friends to server page	19
Delete friends from server page	20
Unified Modelling Language Diagrams	21
Hierarchical diagram	21
Control flow graph	22
Class diagram	23
Sequence diagram	24
Sign up	24
Login	24
Main page	25
Send friend request	25

Send message to a friend	26
Create a server	26
Add friends to server	27
Delete friends from server	27
Entity relationship diagram	28
Object model	29
References	30



Introduction

In this document I will be outlining the design and development plan for Kyle Connect. Kyle Connect's main aim is to allow people to communicate with other people around the world. Throughout this document, I will present the technical aspects involved in the development of Kyle Connect, focusing on key elements such as user experience, user interface design, security, speed, and usability. By prioritizing these aspects, Kyle Connect aims to provide a reliable and intuitive communication platform, ensuring users can connect and communicate with ease and confidence.



Technology Stack: Implementation languages

Frontend languages

During the development of the frontend of Kyle Connect, I will be utilising the use of HTML and CSS. The technologies will form the core structure, design and interactivity of Kyle Connect.

HTML: Hyper Text Markup Language

HTML will serve as the foundational structure for displaying information to users on Kyle Connect. It plays a crucial role in shaping the user interface and enhancing the overall user experience.



CSS: Cascading Style Sheets

CSS will be used to style the HTML pages, creating an aesthetically pleasing and visually engaging experience for users on Kyle Connect. It will enhance the overall look and feel, contributing to an intuitive and enjoyable user interface.





Backend language

During the development of the backend of Kyle Connect, I will be using Go. Using this programming language will form the core processing, server-side logic and data management ensuring fast and reliable performance. Its scalability and efficiency make it an ideal choice for handling Kyle Connect's backend operations.

Go: Golang

I will be using Go due to its high performance and simple syntax and its concurrency support with the use of go-routines. Go is a very good language for building scalable, efficient and very fast processing services.



Deployment

Docker



What is Docker?

Docker is a piece of software that allows developers to combine all of their work together and distribute this work to their customers in a very portable manner. This allows the developers to combine all of the components needed for the project together such as, your code, libraries, dependencies, run-time and much more. When all of these components are combined together this is called a Docker Image.



After a Docker Image is executed it is put into a Docker container, this is an isolated part of the application. A Docker container shares the operating system's kernel but this remains independent of each other, this means that the application can work on any machine such as Windows, Mac, Linux. This eliminates the term that is commonly known in software development as "it works on my machine".

Why am I using Docker?

The main reason why I am using docker is for the following reason. At the start of the development of Kyle Connect, in order for me to see if my signup code or login code was working or not I would have to type in a command. For example, `go run path/main.go`.

This command would run my go code for me. But typing in this command every time was quite time consuming and not convenient for me. So I did some research and during my research I found a piece of software called Docker, then I went off and watched some YouTube videos on what docker is and how to get it installed.

When I was watching some of the videos on Docker [6] I thought, would it be possible for me to be able to use Docker to run my go code for me automatically. Docker turned out to be a perfect solution. By making a Dockerfile, I was able to specify all of the steps needed to run my application, this included the building and the execution of my go code.

Now, instead of having to manually type in a command to run my code repeatedly, I can simply use a docker container to run my application with my go code. This means that docker has saved me some time but it also ensures that my application runs consistently, and this will work in any form of environment.



Algorithms

What are algorithms?

An algorithm is a procedure used for solving a problem or performing a computation. Before I started to code out my project I was anticipating needing to use one or many algorithms for the development of Kyle Connect.

The types of algorithms I need for Kyle Connect are encryption algorithms, due to Kyle Connect being a communication system and I must ensure the security of any user that uses Kyle Connect.

During my research I found out that there is a library in go for using encryption algorithms. In the go standard library I found something called “bcrypt” [1]. Bcrypt uses the Provos and Mazières's hashing algorithms [2].

I am currently using the “bcrypt” library to use two of the algorithms in that library called CompareHashAndPassword and GenerateFromPassword.

The CompareHashAndPassword algorithm takes in plaintext and encrypts the plaintext. The GenerateFromPassword algorithm takes in the encrypted text and decrypts it to the original plaintext.

At the moment I am only using two of the algorithms in the “bcrypt” library but I might need to use more of these algorithms further in the development of Kyle Connect.



Pseudocode

Pseudocode for CheckPassword algorithm:

```
CheckPassword(password, hash type) {  
    decryptedPassword = invoke CompareHashAndPassword(byte  
array[hash], byte array[password])  
    return decryptedPassword  
}
```

Pseudocode for HashingPassword algorithm:

```
HashingPassword(input type) {  
    hash, err = call GenerateFromPassword(byte array[input],  
defaultCost)  
    // do some error checking  
    if err is not equal to null {  
        return "" and err  
    }  
    return from method == type(hash)  
}
```

PseudoCode for GenerateFromPassword Algorithm:

```
GenerateFromPassword(hash []byte, cost int) []byte, error {  
    If the length of the password is greater than 72 {  
        return nil, ErrPasswordTooLong  
    }  
    return password.Hash(), nil  
}
```

PseudoCode for CompareHashAndPassword Algorithm:

```
CompareHashAndPassword(hashAndPassword, password []byte) error {  
    return ErrMismatchedHashAndPassword  
}
```




Database

My-SQL

I have chosen to use my-sql for my project because that is the most familiar database that I am used to working with. Due to my project being a communication system it makes more sense to use a structured database system with rows and columns. For example, storing username, email and password. This means it is easy to write an sql query to retrieve information from my database. My-sql is a relational database system (RDBMS). Sql stands for Structured Query Language [7]. A relational database stores data in tables, which the data might be related to. Sql is a common language that developers use when creating applications, this is due to being able to create, modify and extract data from the database.




User-interface design

Wireframe

Welcome page

Welcome to Kyle Connect

[Send Message](#)

YES NO

New to Kyle Connect?


Create an account

Already have an account?

Login

Sign up

Sign up

[Send Message](#)

YES NO

Name:

Email:

Password:

Login

Login

Email:

Enter your email

Password:

Enter your password

Login

Send Message

✓

✗

YES NO



Main page - user interface

Actions	Welcome, Kyle	Friends	Pending Friend Requests
			Harry <div>Accept</div> <div>Decline</div>
<div>Add friend</div> <div>Create a server</div> <div>Add friend to server</div>	<div>Kyle</div> <div>Hi Joe</div> <div>Joe</div> <div>Hi Kyle</div>	<div>Barry</div> <div>Jerry</div> <div>Kyle</div> <div>Joe</div>	
	Servers		
	<div>A</div> <div>B</div> <div>C</div> <div>D</div> <div>E</div> <div>F</div> <div>G</div> <div>H</div>		

Send friend request

Add a friend

Enter a name of a friend that you would like to add

Username:

Send Message

☒ ☐

YES NO

Send message to a friend

Send a Message to a friend

Your message:

Send Message

☒ ☐

YES NO

Create a server

Create a server

What do you want to call your server?

Server name:

Send Message

☒ ☐

YES NO

Add friends to server

Add friends to your server

Click on what friends that you want to add to what server.

Friends

☐ Joe

☐ Bob

☐ Kyle

☐ Barry

Servers

☐ Movies

☐ Sports

☐ Games

☐ Coding

Send Message

☒ ☐

YES NO

Delete friends from server

Delete friends from server

Click on what friends that you want to delete from what server.

Send Message

✓

✗

YES NO

Friends

Joe

Bob

Kyle

Barry

Servers

Movies

Sports

Games

Coding

Delete friends from server



Target platform

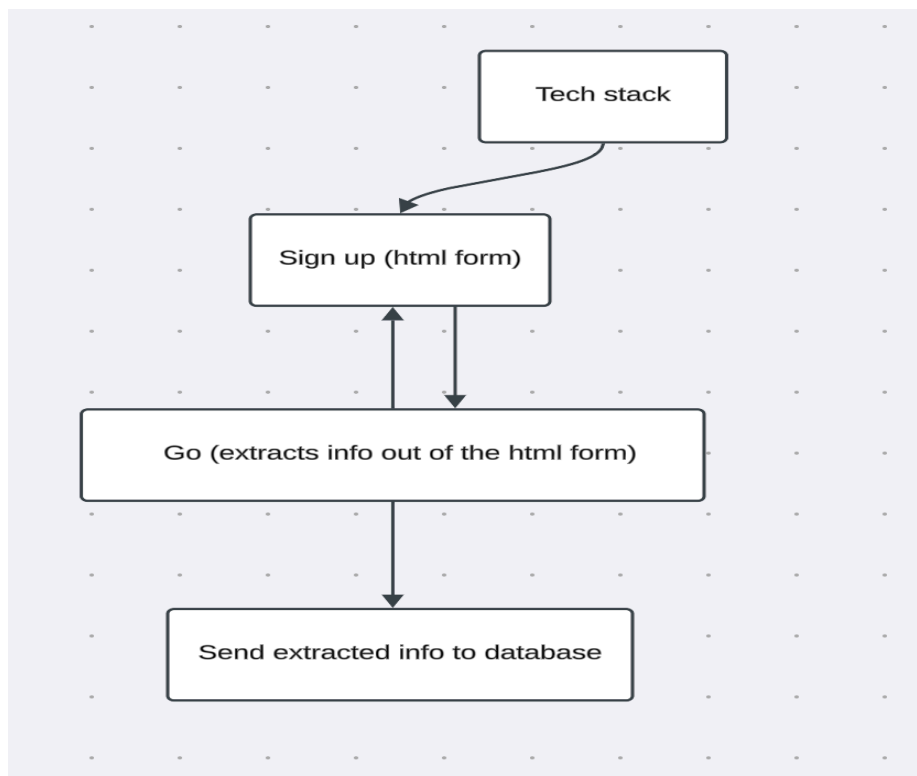
The target platform for my users to use my product is going to be a website. I chose to pick a website because it is very accessible for anyone to use. Developing a website ensures that my project is cross-platform meaning users on different operating systems such as (Windows, Mac and Linux) and Browsers such as (Firefox, Chrome) can all access my project without any potential issues.

Additionally it is easier to update and maintain, as changes can be put out right away to all users without problems occurring. This makes the website a good choice for keeping Kyle Connect up to date with the latest features and much more.



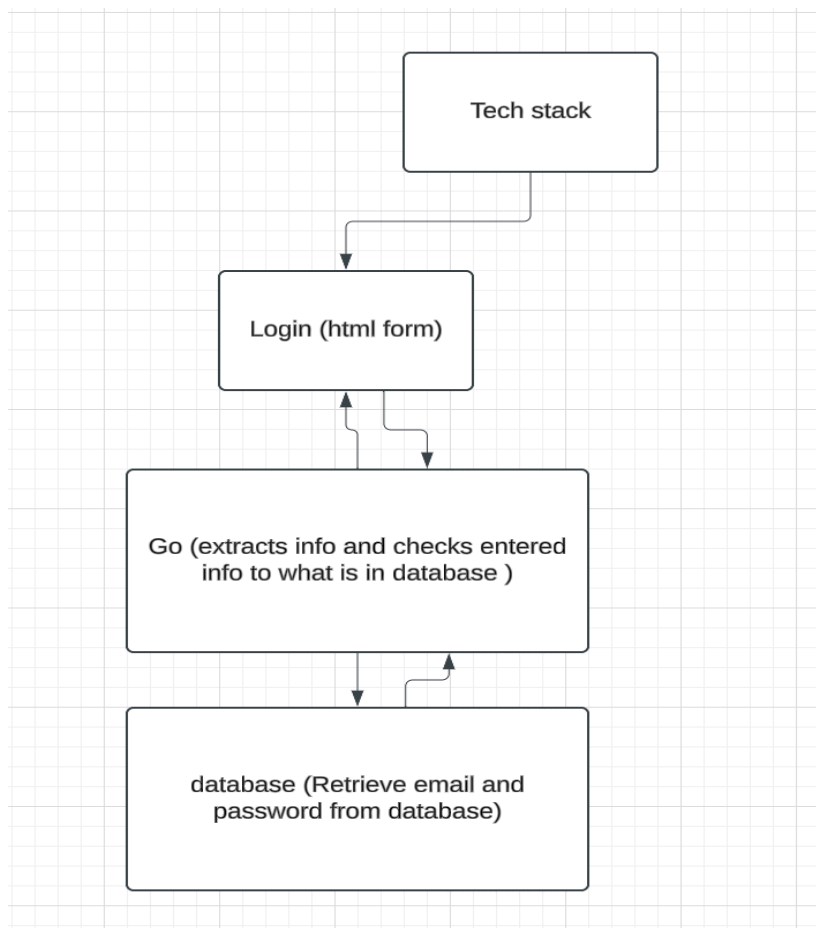
How Technology Stack communicate together

Sign up page



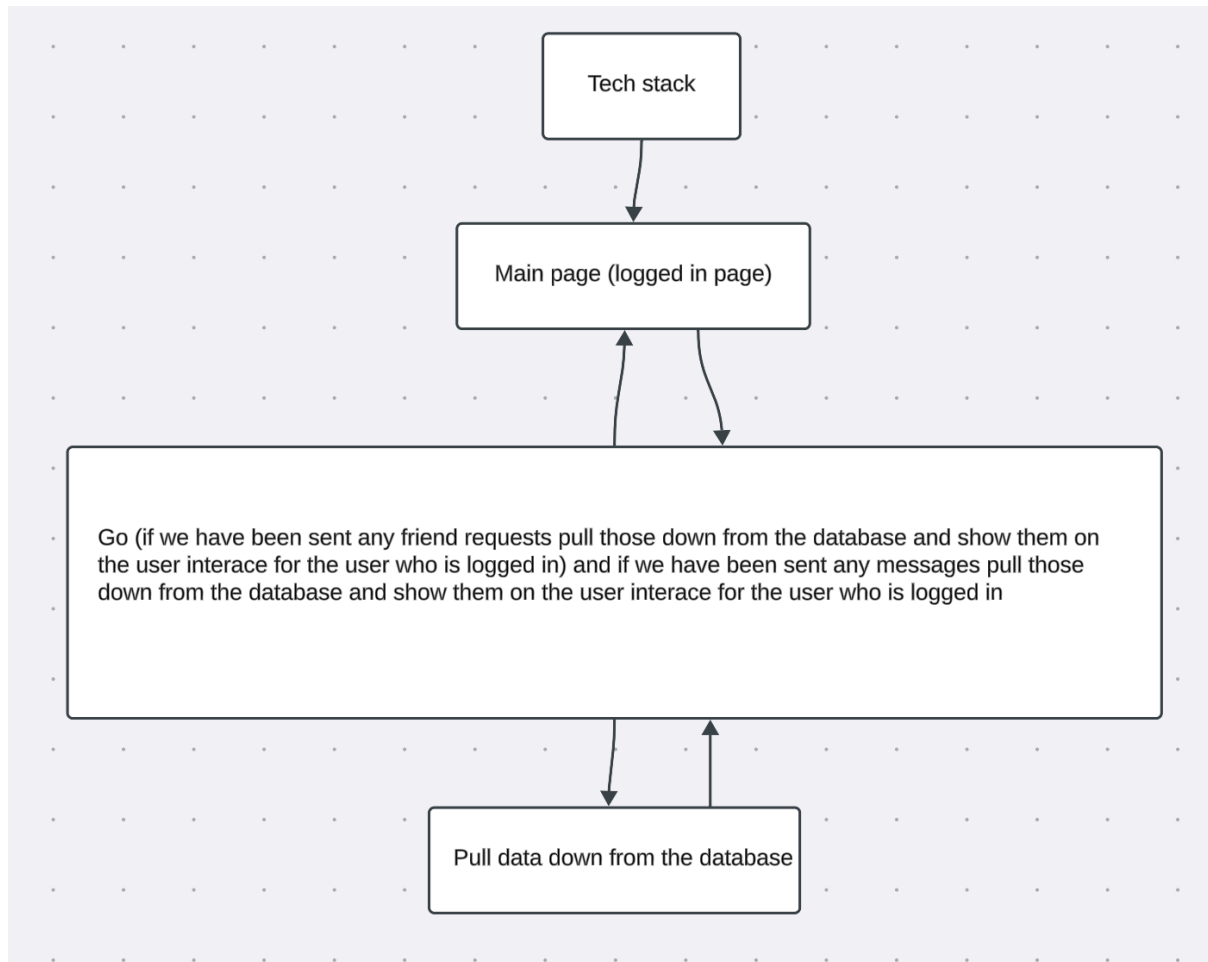


Login page



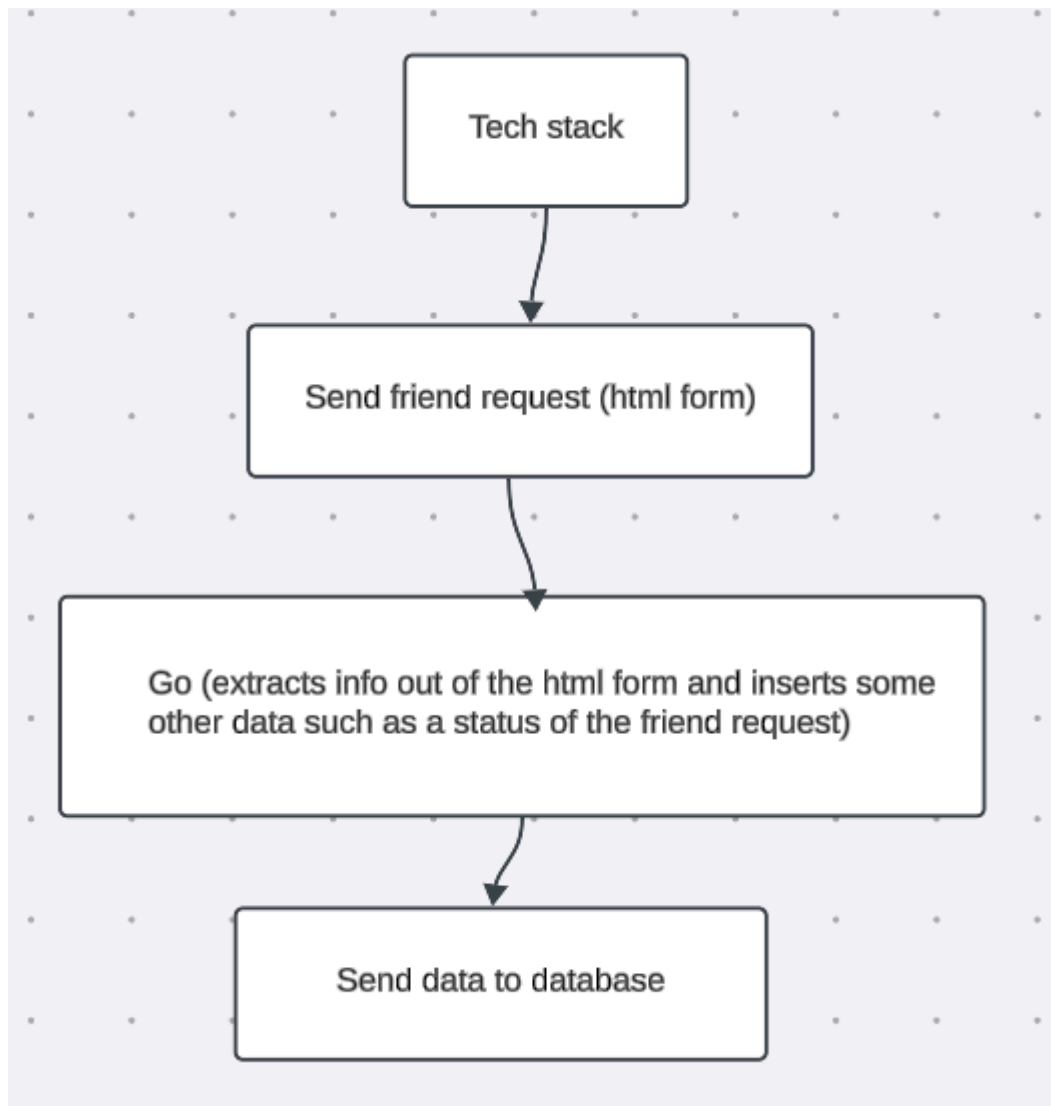


Main page - user interface



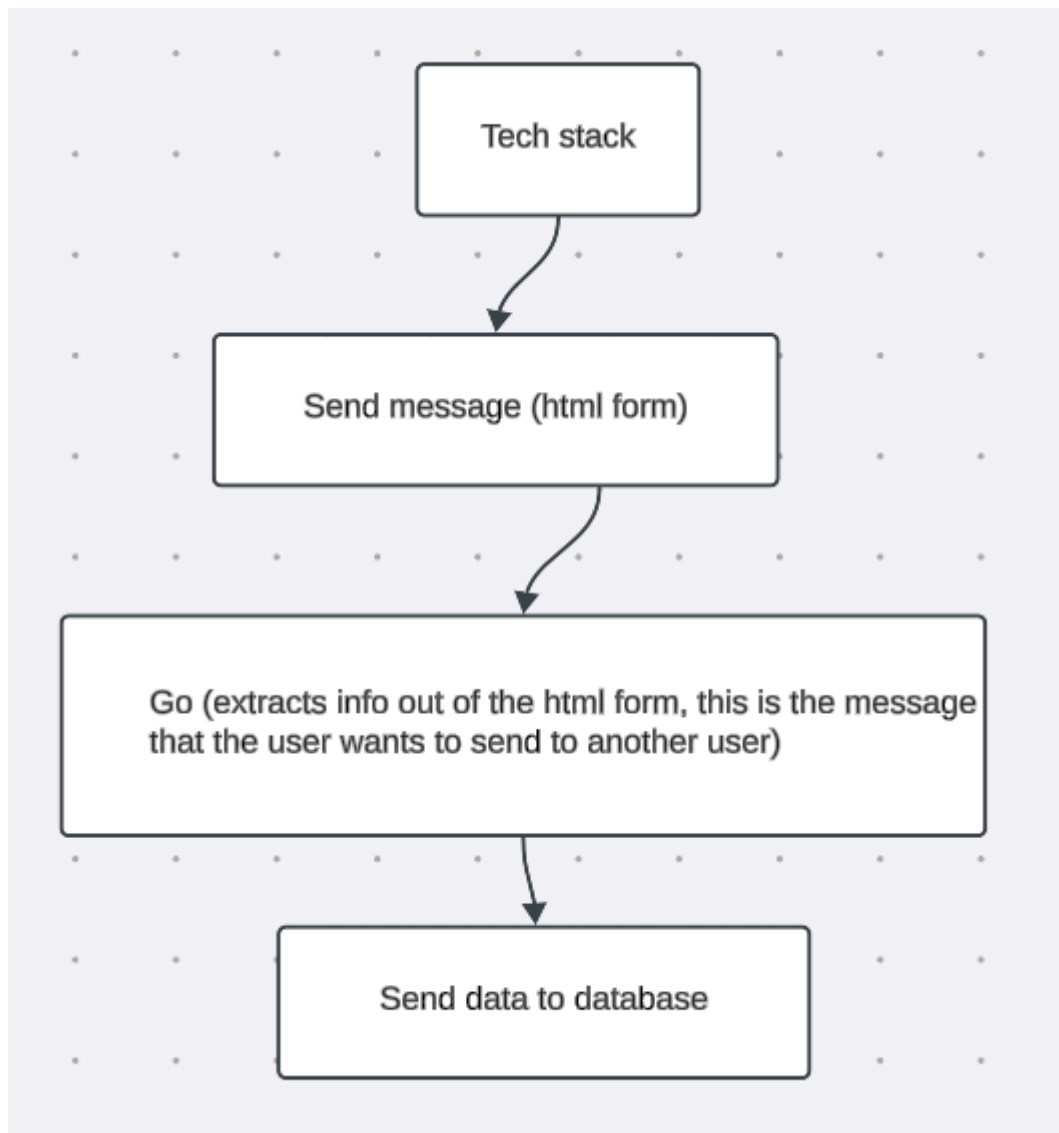


Send Friend Request page



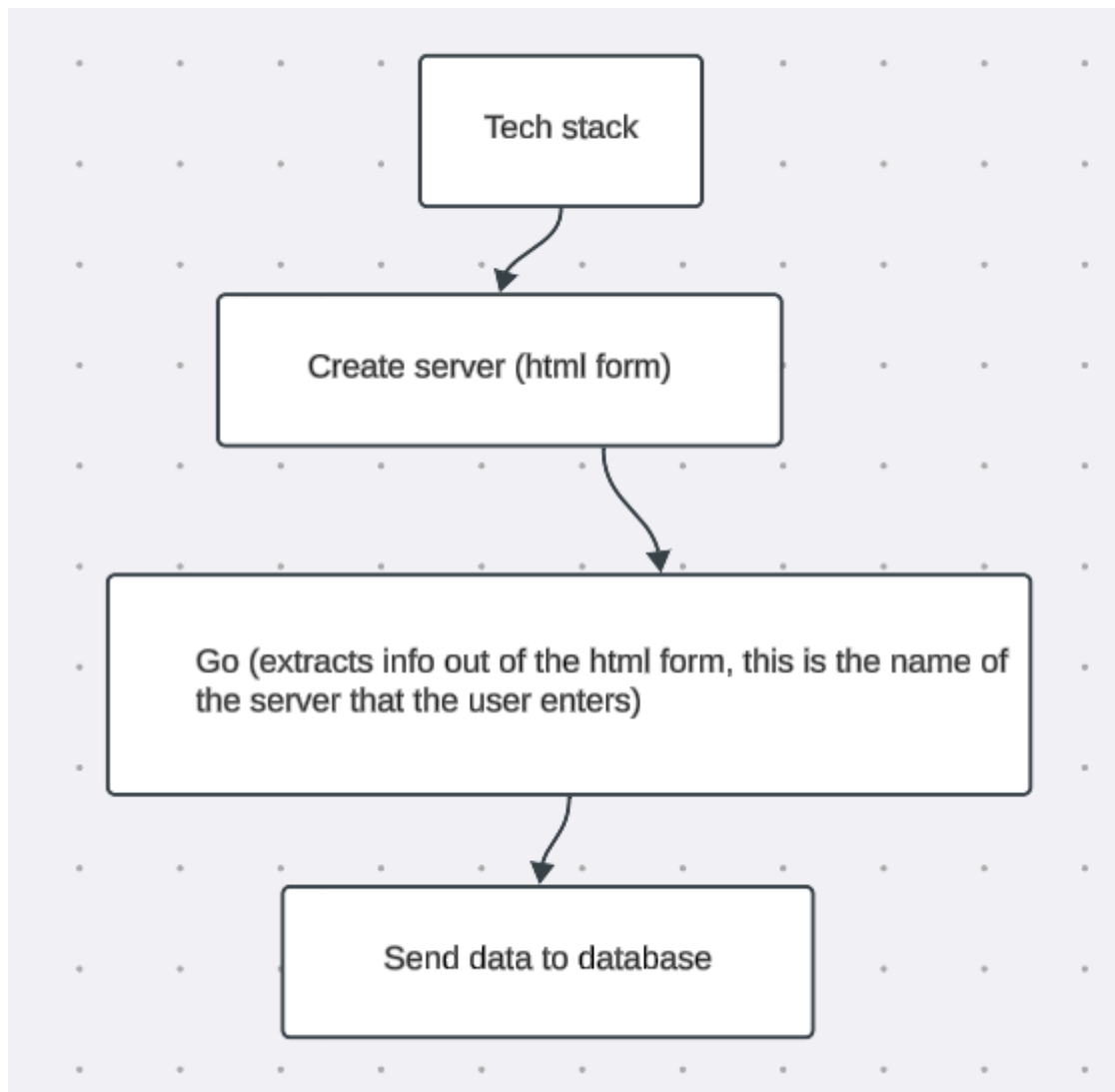


Send message to a friend page



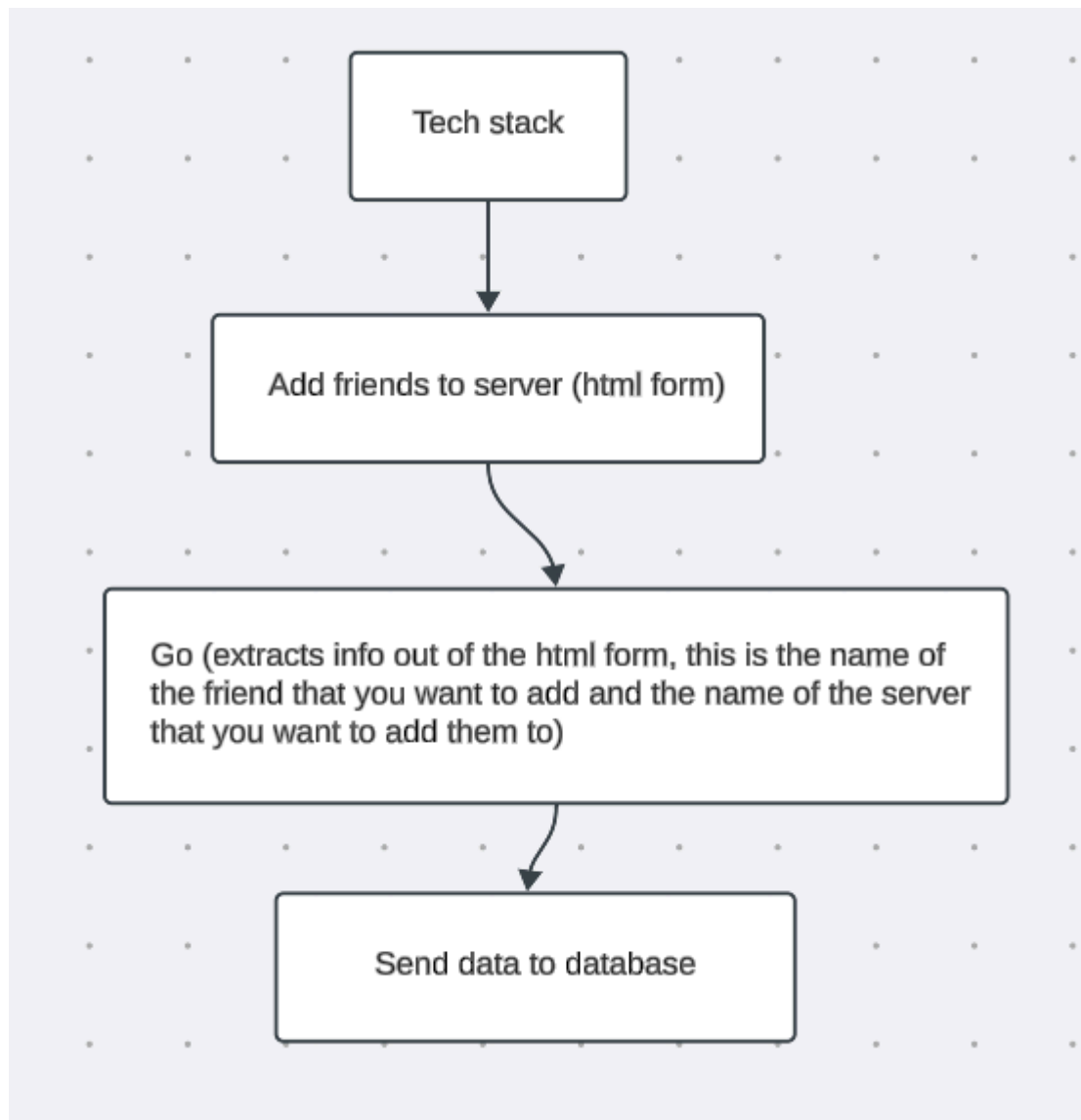


Create a server page



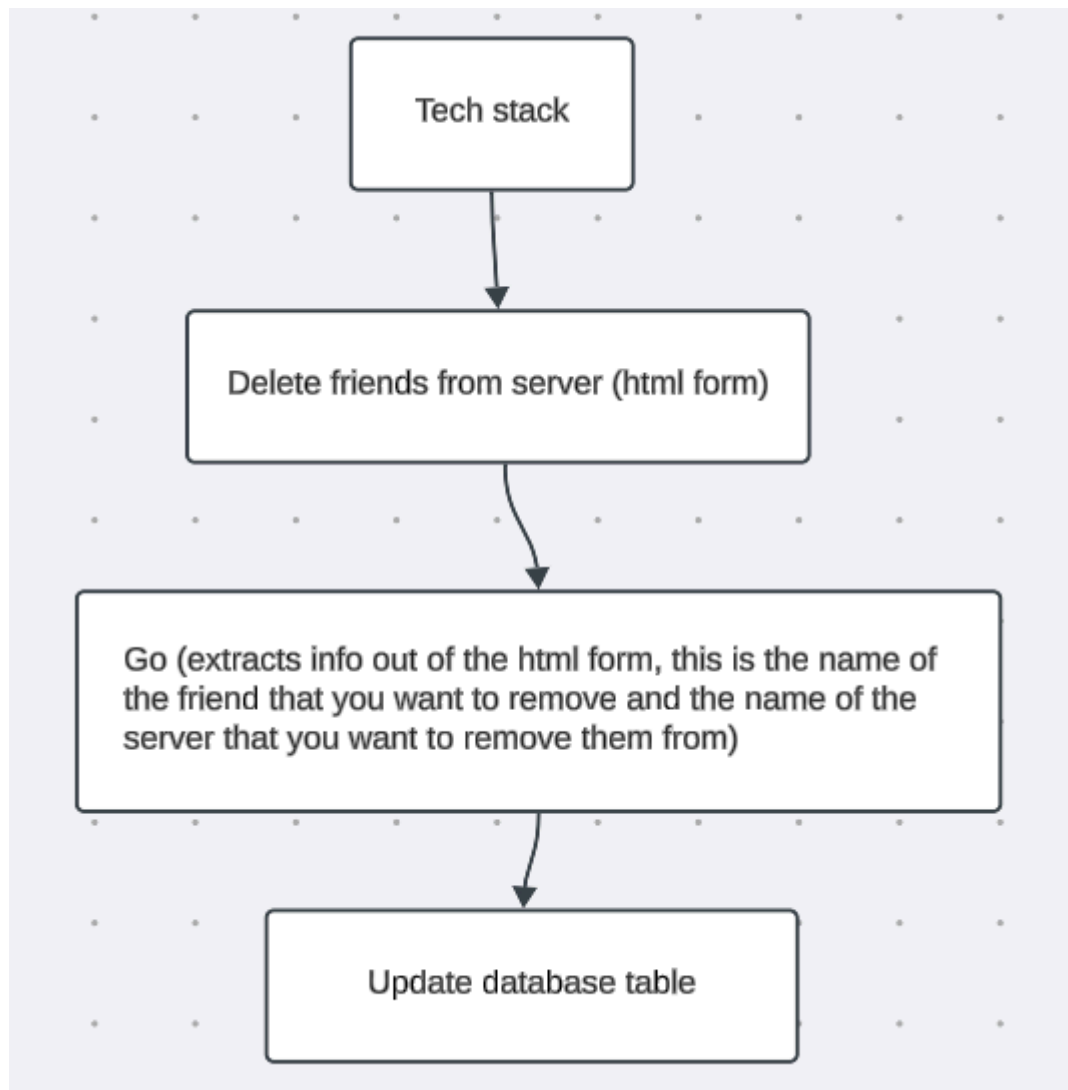


Add friends to server page



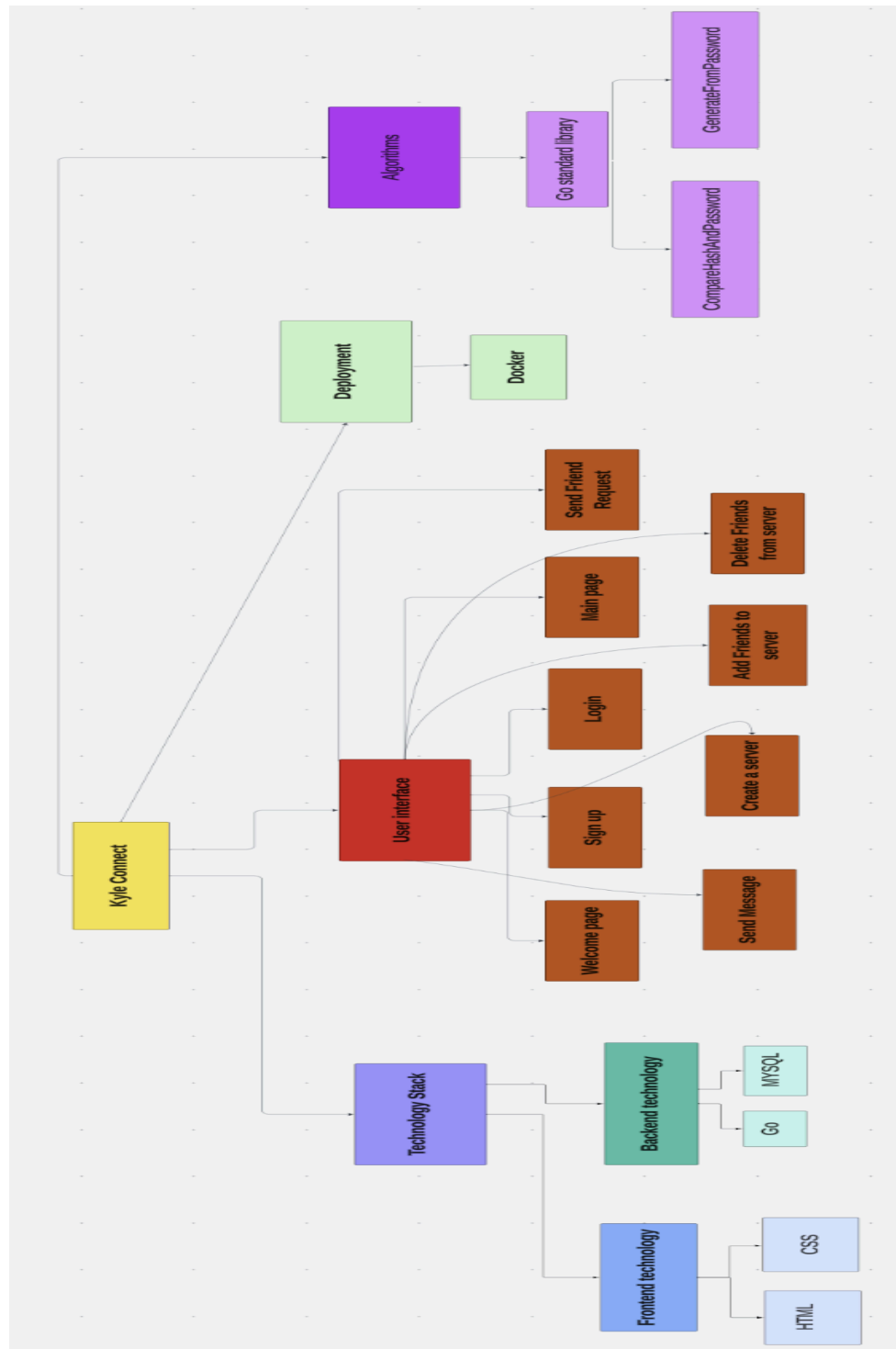


Delete friends from server page



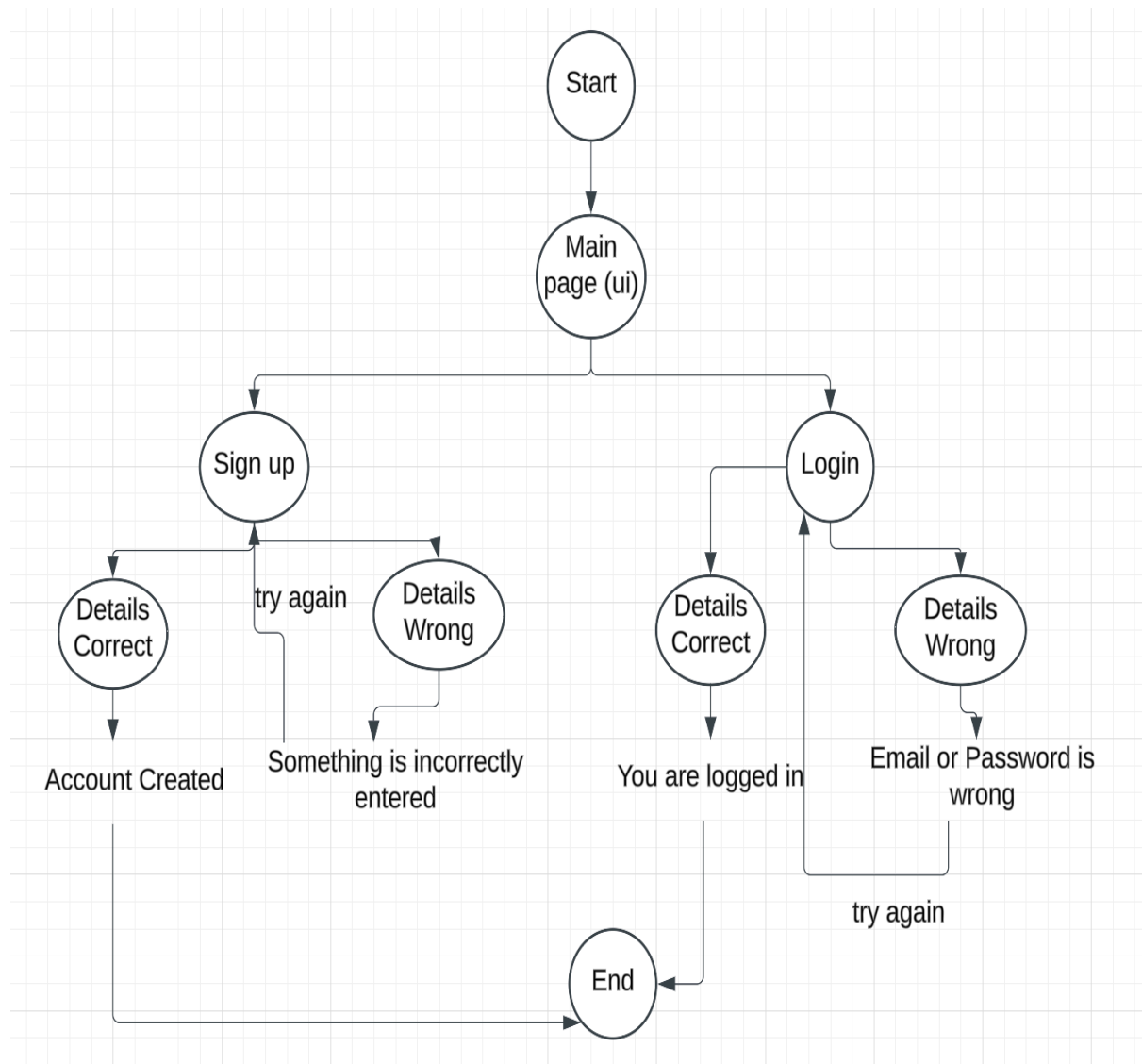
Unified Modelling Language Diagrams

Hierarchical diagram

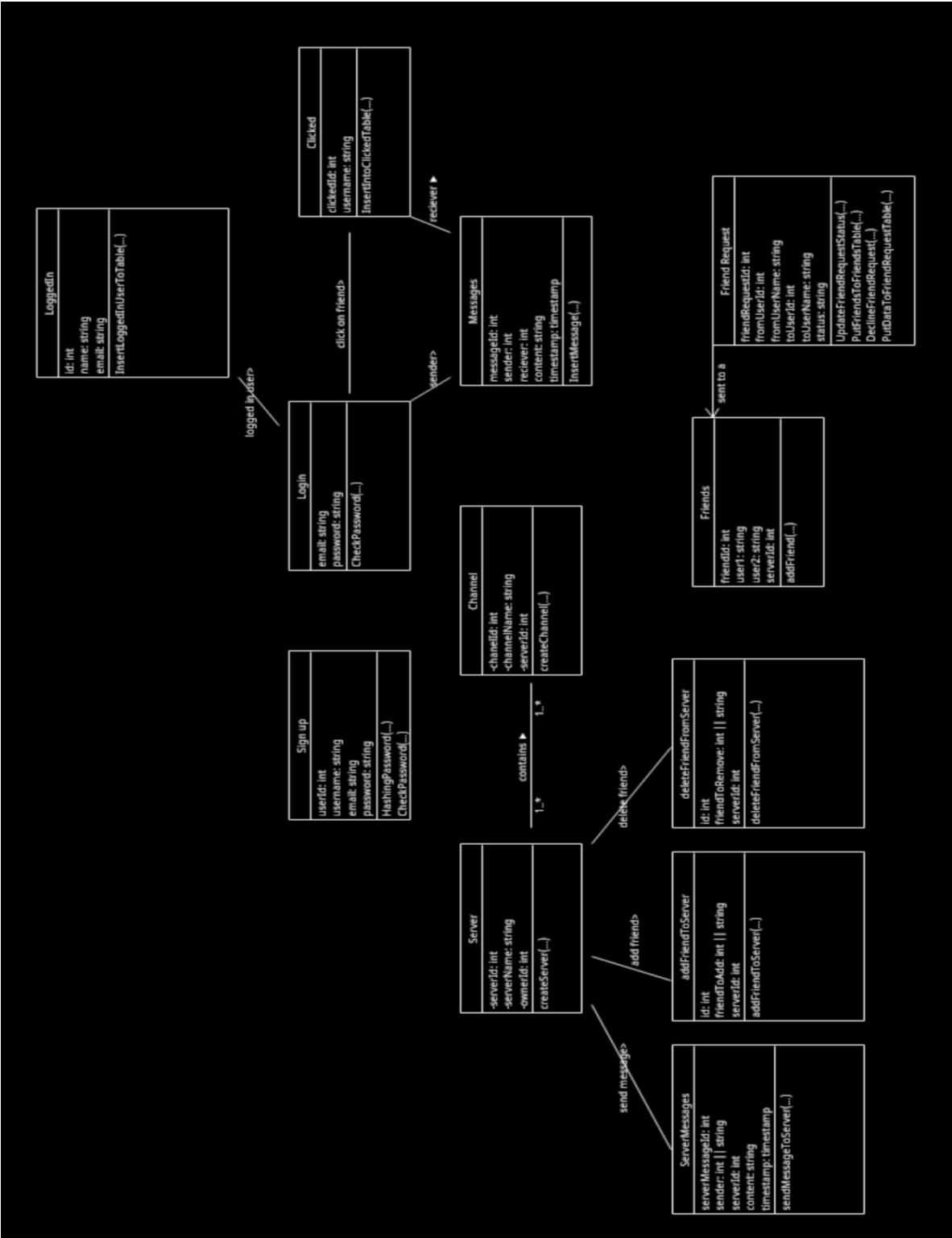


Control flow graph

From main page to sign up or login

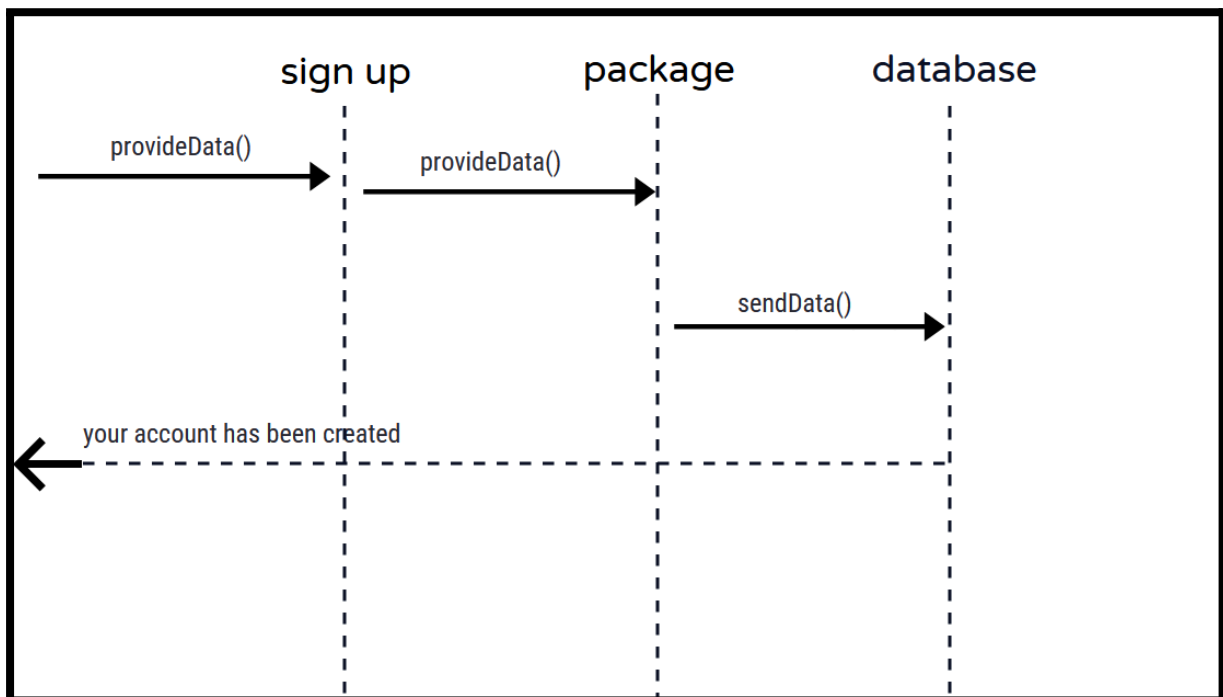


Class diagram

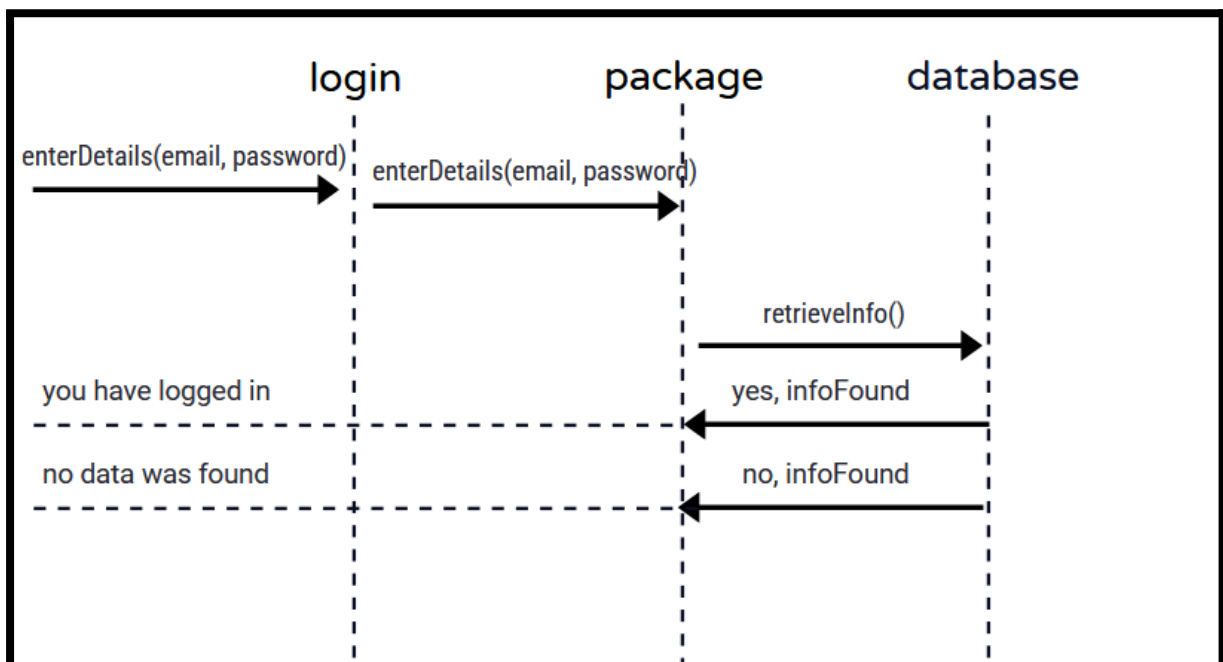


Sequence diagram

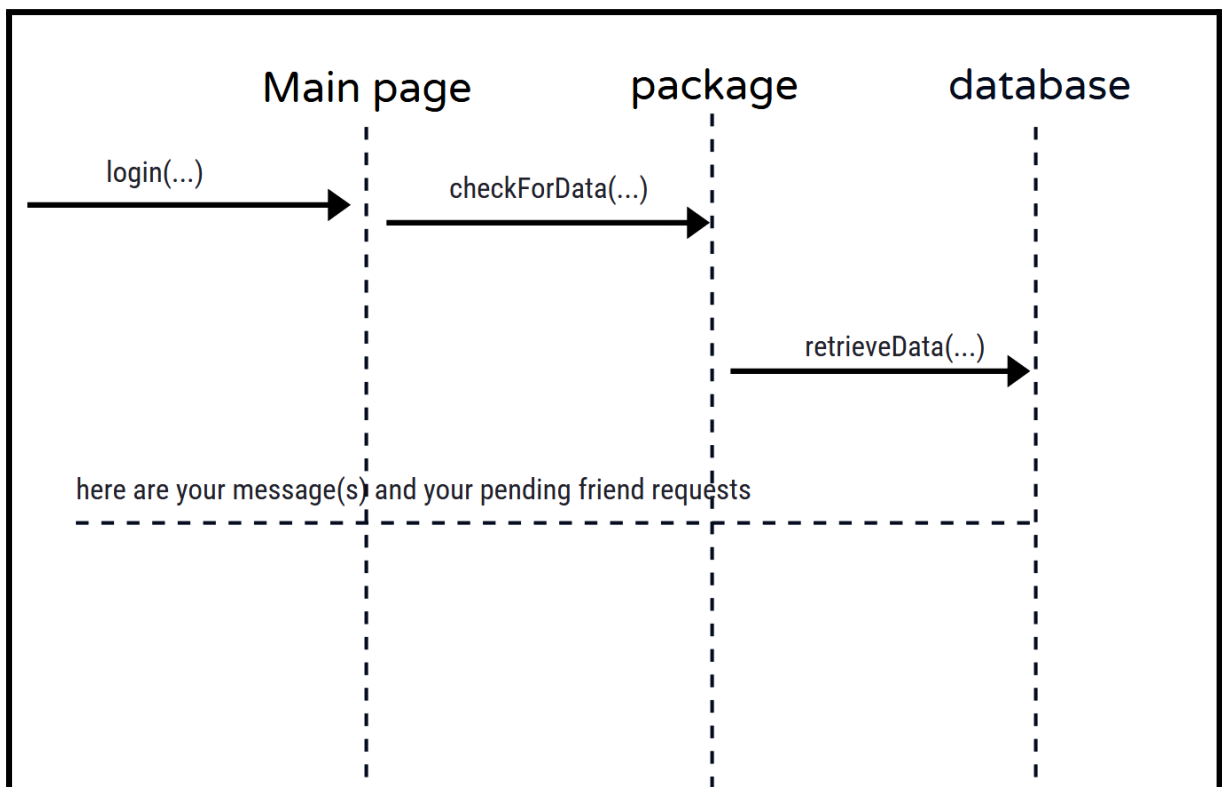
Sign up



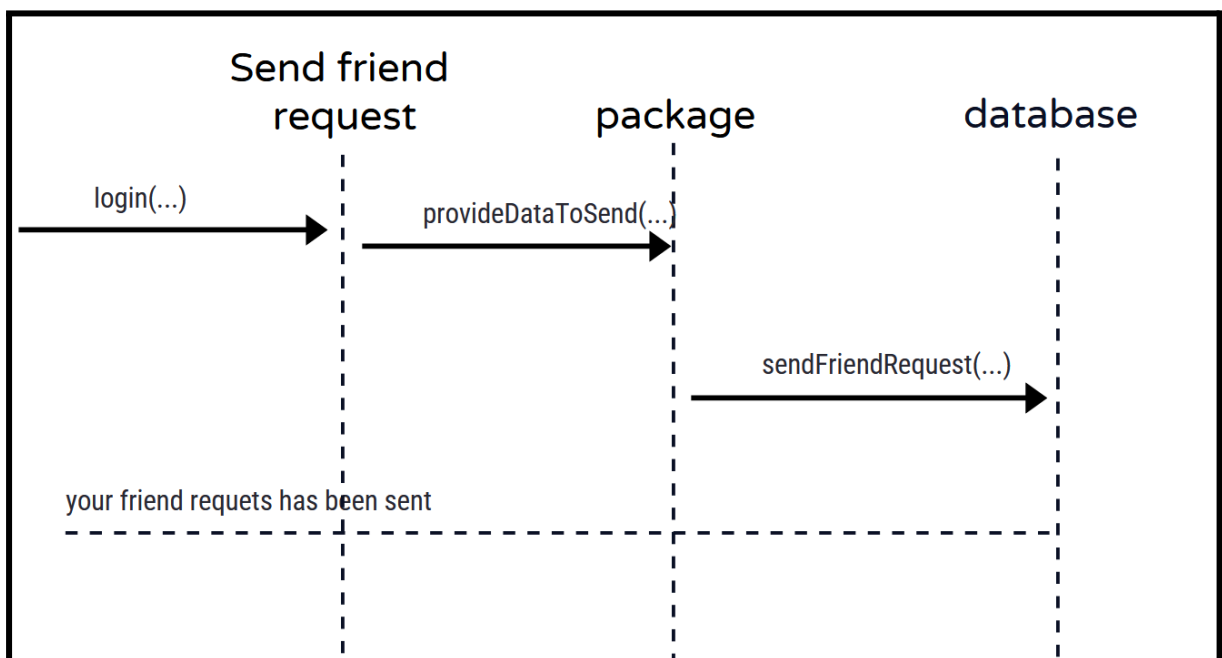
Login



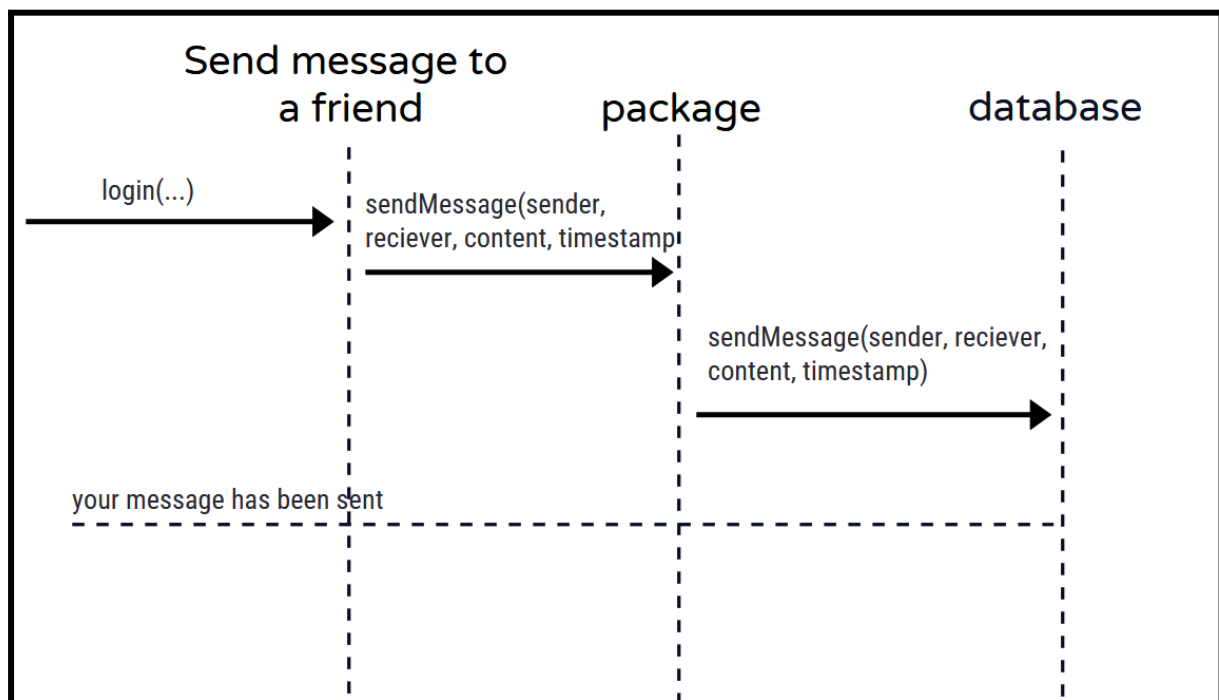
Main page



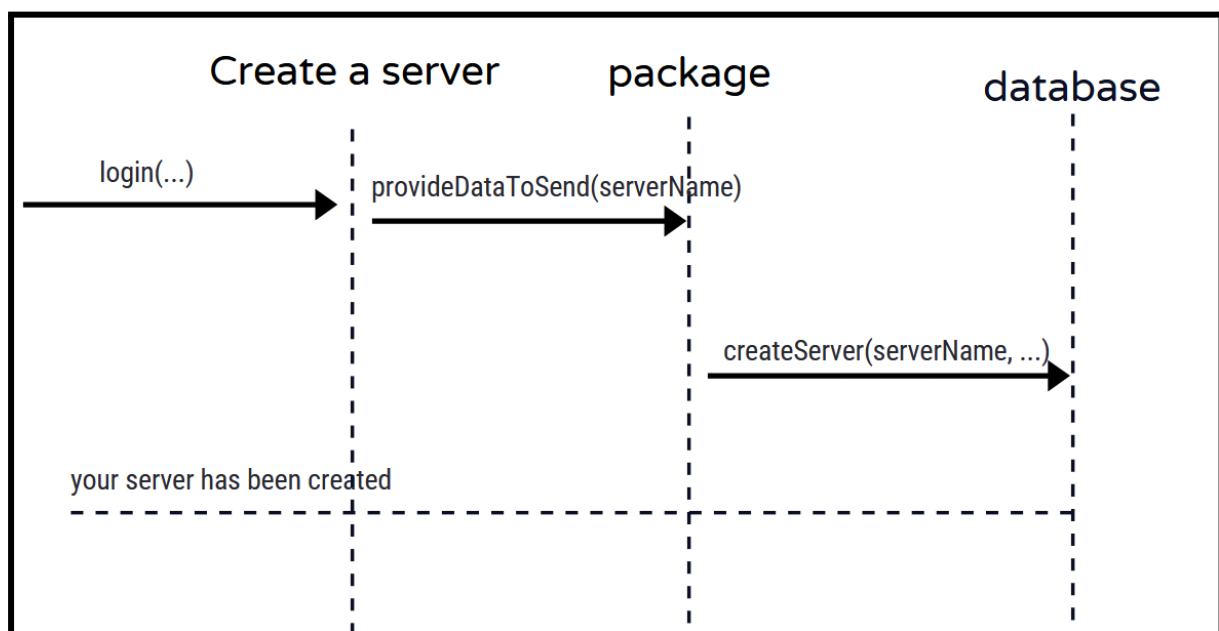
Send friend request



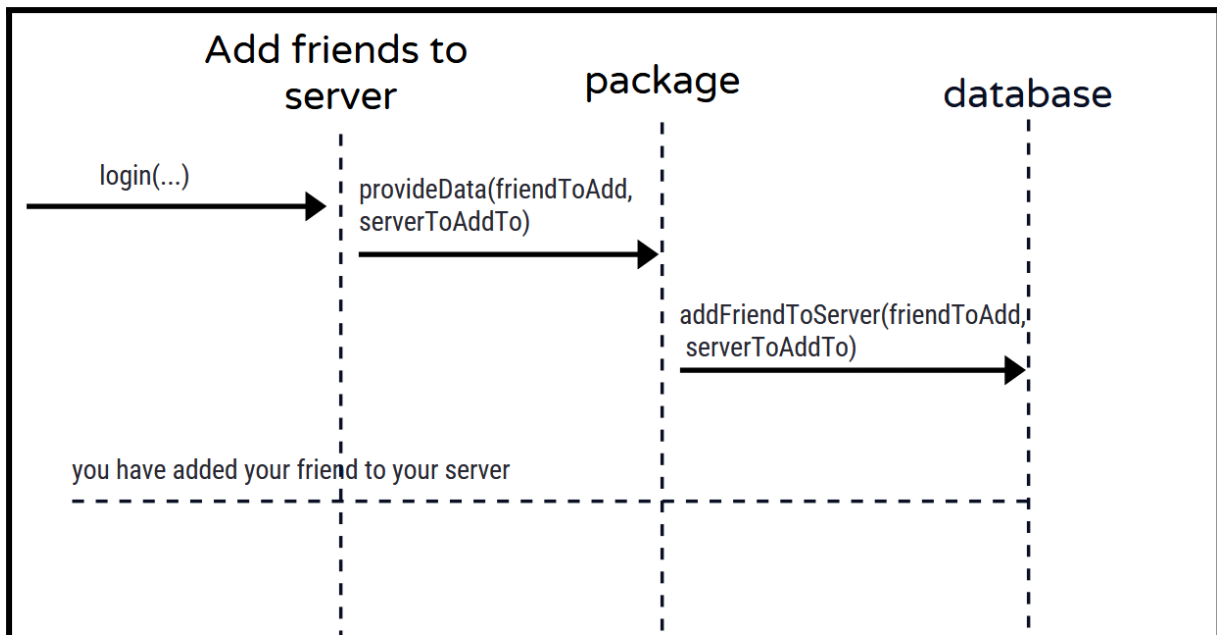
Send message to a friend



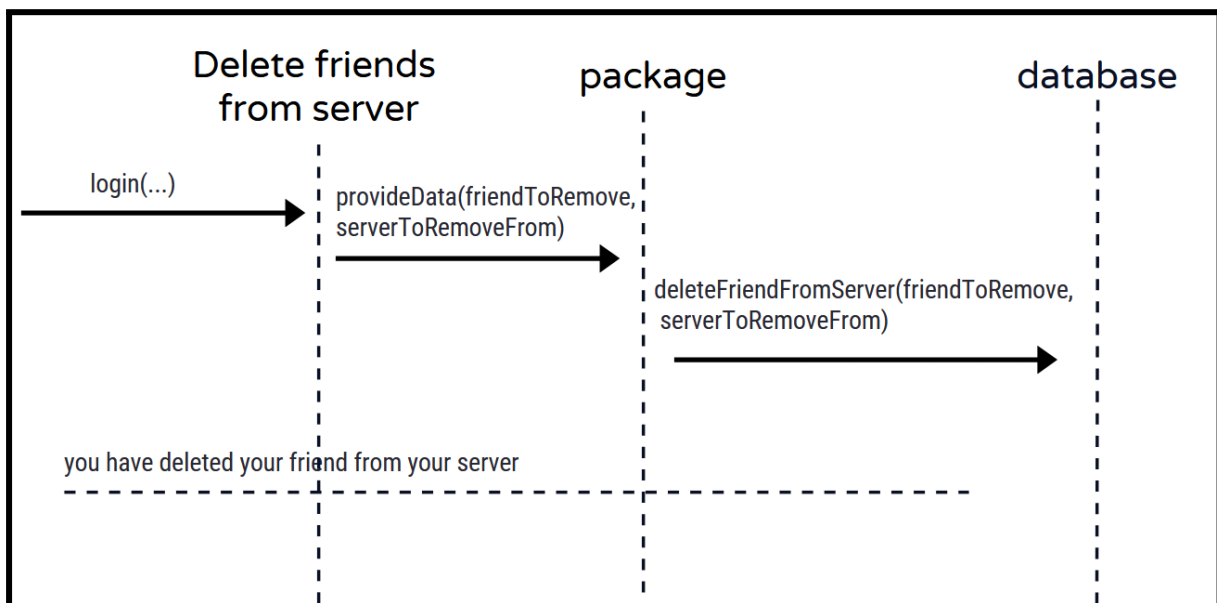
Create a server



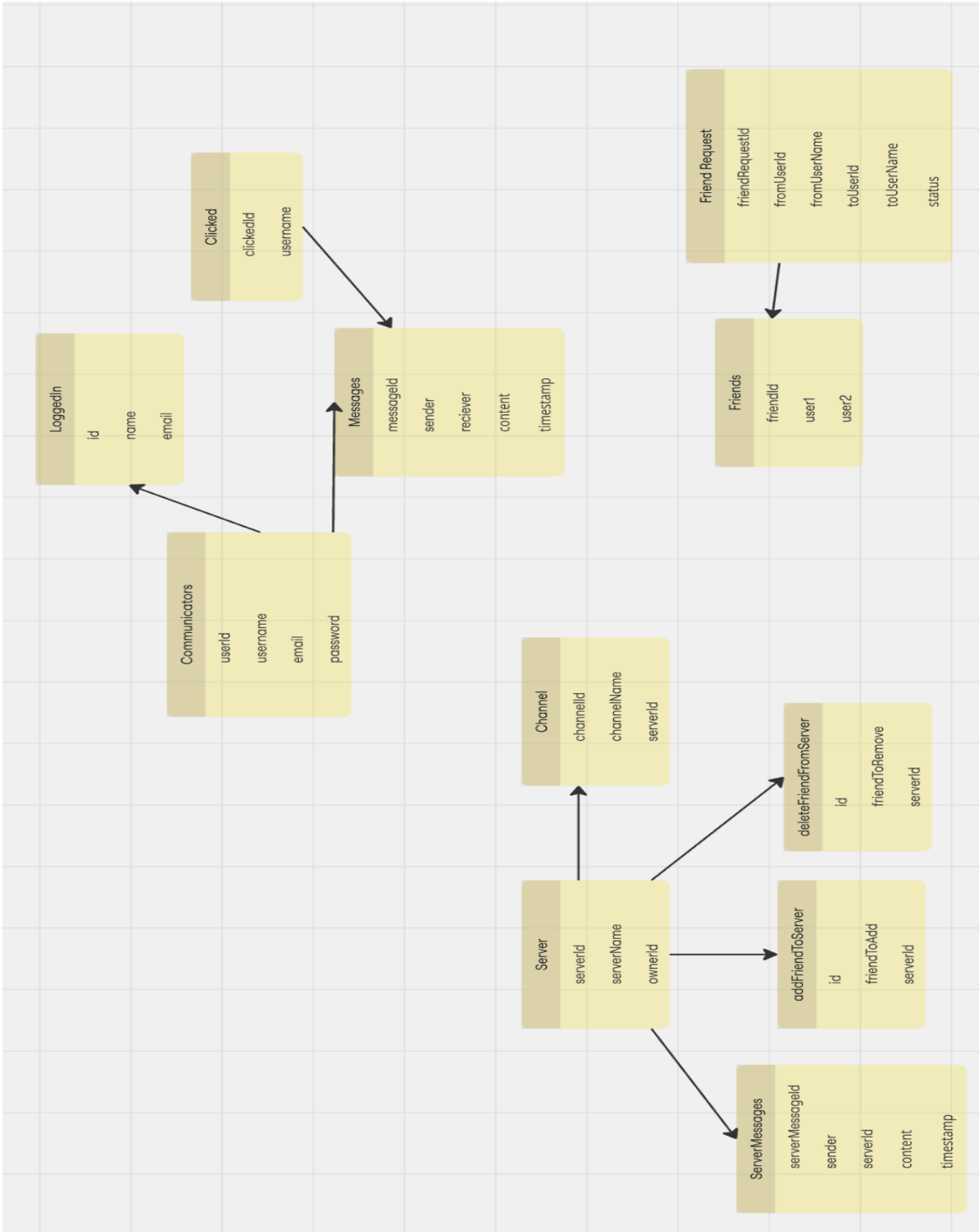
Add friends to server



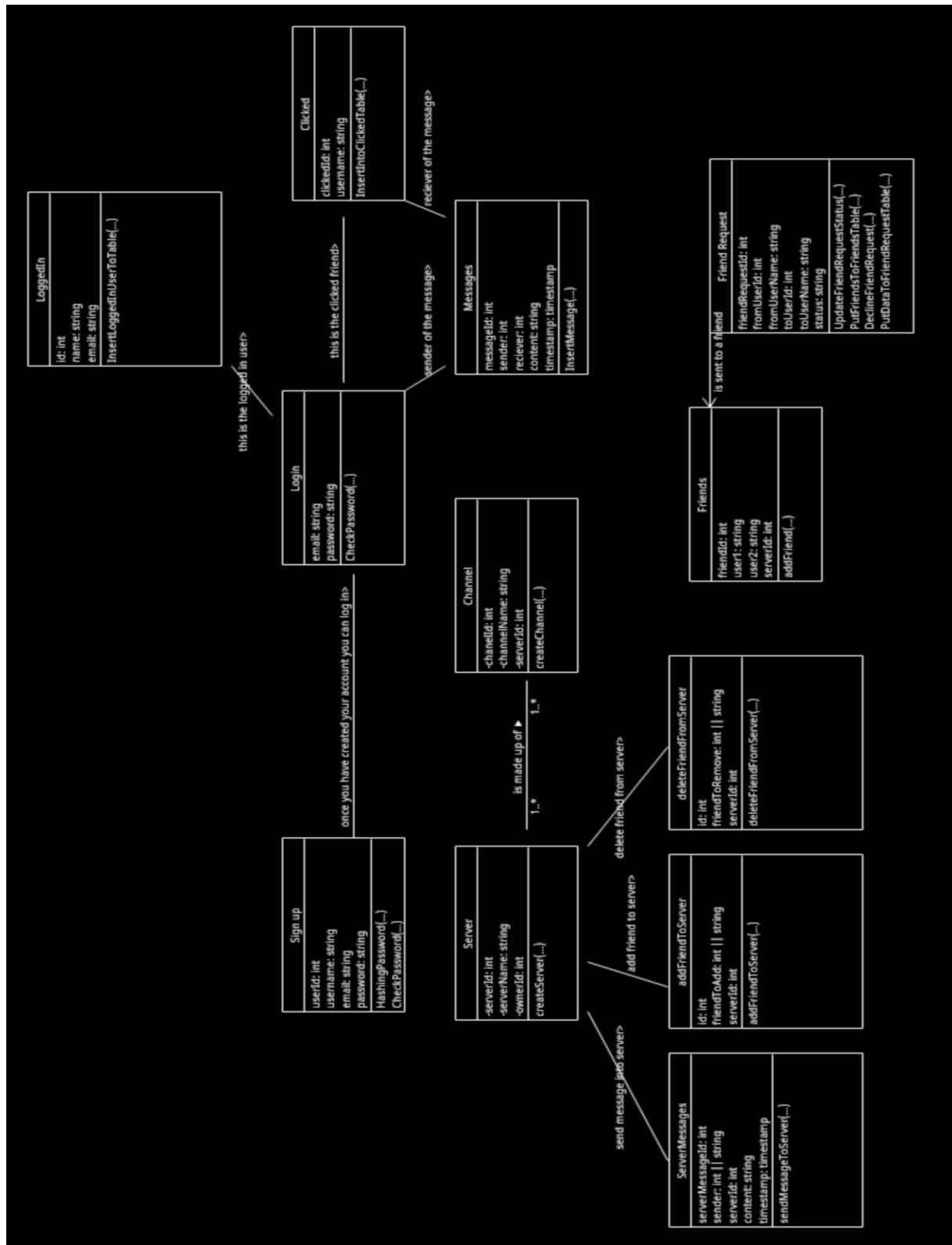
Delete friends from server



Entity relationship diagram



Object model





References

pkg.go.dev. (n.d.). *bcrypt package - golang.org/x/crypto/bcrypt - Go Packages*.

[online] Available at: <https://pkg.go.dev/golang.org/x/crypto/bcrypt>. [1]

Wikipedia. (2020). *bcrypt*. [online] Available at:

<https://en.wikipedia.org/wiki/Bcrypt>. [2]

www.umletino.com. (n.d.). *UMLetino - Free Online UML Tool for Fast UML*

Diagrams. [online] Available at: <https://www.umletino.com/umletino.html>. [3]

Lucidchart. (n.d.). *Intelligent Diagramming*. [online] Available at:

<https://www.lucidchart.com/pages/>. [4]

Wikipedia.org. (2024). *File:Docke (container engine) logo.svg - Wikipedia*. [online]

Available at:

https://en.m.wikipedia.org/wiki/File:Docke_%28container_engine%29_logo.svg

[Accessed 29 Dec. 2024]. [5]

Techno Tim (2022). *Build YOUR OWN Dockerfile, Image, and Container - Docker Tutorial*. [online] YouTube. Available at:

<https://www.youtube.com/watch?v=SnSH8Ht3MIc&list=PL82FQWeDS43DEnvccwQ-9jaYlf4Dw4TJ-&index=5> [Accessed 29 Dec. 2024]. [6]

Wikipedia Contributors (2019). *MySQL*. [online] Wikipedia. Available at:

<https://en.wikipedia.org/wiki/MySQL>. [7]

