Shaking Hands Overseas

Development of a human anatomy-based hand system

Alex Trias Betorz

Eduard Lleget Viladés

Joel Garcia Martin

Martí Viñolas Parcet

Curs 2021-2022

Mentored by Alicia López

CIC Escola de Batxillerats

10th of April 2022

Contents

Abstract

English

In days of quarantine during last year’s pandemic, a new human necessity was perceived: The need for human contact. From that need we started developing the idea of how amazing it would be to interact with people without the need to share the same space, and this project will aim to recreate this interaction through the simplest human act of kindness: Shaking Hands. We will cover all the aspects and concepts that occur during this kind of interaction and what actually happens in the background while we shake hands. And with it, we aim to understand how the internet is tied together, how servers work and how to recreate the complexity and beauty of a human hand and forearm. This is the result of 8 months of hard work and dedication to accomplish an astonishing goal, to shake hands overseas.

Catalan

En dies de quarantena, durant la pandèmia de l’any passat, es va percebre un nova necessitat humana: la necessitat del contacte humà. A partir d'aquest principi vam començar a imaginar-nos com d’al·lucinant seria poder recrear aquesta interacció sense la necessitat de la presència física. Aquest projecte recrearà aquesta interacció amb el més simple però més sincer acte de bondat: donar-se la mà. Passarem per tots els aspectes i conceptes que son necessaris per a aquest tipus de connexió i pretenem entendre com la internet funciona, com els servidors funcionen i com recrear la complexitat i bellesa d’una mà i avantbraç humà. Aquest és el resultat de 8 mesos de treball i dedicació per intentar complir un objectiu insòlit, donar-se la mà a través de l’oceà.

Spanish

En días de cuarentena, durante la pandemia del año pasado, percibimos una nueva necesidad humana: La necesidad del contacto humano. A partir de este principio, empezamos a imaginarnos que alucinante seria poder recrear esta interacción sin la necesidad de la presencia física. En este proyecto intentaremos recrear esta interacción con el más simple, pero más sincero acto de bondad: Dar-se la mano. Pasaremos por todos los aspectos i conceptos necesarios para este tipo de conexión y entender cómo funciona el internet, los servidores y como recrear la belleza i complejidad de un brazo y antebrazo humano. Este es el resultado de 8 meses de esfuerzo y dedicación para intentar cumplir un objetivo insólito, darse la mano a través del océano.

1. Introduction

1.1 Background

Few would dispute the fact that the need for updated prosthesis has started to expand, reaching from legs to eyes. It is not vital either to build them or to build airplanes, meaning that humans are constantly in the research for knowledge, that is what drives science and society itself. In any case, there is no deficiency of request or potential applications for both, but the prosthesis built so far do not meet the expectations on what people and creatures can achieve. This can be especially genuine in control, where mechanical hands do not approach the tangible engine capacity of human hand.

A substantial portion of human mechanical interactions with the environment are performed by the hands. They allow us to perform altogether different tasks, from applying high forces (e.g., utilizing a sledge) to executing precision tasks (e.g., handwriting). They have a conceivable range of motion due to a complex constitution: an incredible number of bones associated through various joints, a dense musculature, and a vast sensory system. This range of movements are inherent of the humans' species and trying to recreate them is one of the most challenging aspects of human knowledge evolution.

Biomechanics is the study of the structure, function, and movement of the mechanical aspects of biological systems, using the methods of mechanics leads to numerical portrayals that are used to perform quantitative examinations on this system, these are called biomechanical models of the hand. These portrayals show varied components of the hand which are characterized as far as unbending bodies, joints, and actuators, and therefore the mechanical laws are applied. As they are worked on numerical models of the reality, their utilization and legitimacy rely upon the disentanglements considered. The first biomechanical models of the hand were created to clarify constantly the usefulness of varied anatomical components. In such manner, we can discover numerous works that contemplated the capacity of the natural muscles, Leijnse and Kalker [1]. These models were, however, extremely restricted -two-dimensional- just permitting the investigation of flexion-augmentation developments, they displayed only 1 finger, and that they included significant rearrangements. Moreover, hardly any three-dimensional models had been created around the 21st century, Biryukova and Yourovskaya [2] being an exception, and none of them demonstrated the whole hand. Since 2000, numerous three-dimensional biomechanical models were found in writing, having been produced for altogether different purposes, Fok and Chou [3]: going from grasping the work of the distinctive anatomical components to re-enact ligament movement and joint substitution for medical procedures, to plan prosthetics and biomedical inserts.

A promising exploration region lies ahead with the researcher, desiring to acquire a more complete model of the hand, incorporating information and advancements from the fields of biomechanics, ergonomics, mechanical technology, and PC liveliness. Once we connect these biomechanical models to this last one, the vast internet net, we can expect a relationship that stays for the eternity, in which we fell was perfect as a matter of study.

[1] Freitas, S. R., Mendes, B., Le Sant, G., Andrade, R. J., Nordez, A., & Milanovic, Z. (2017). Can chronic stretching change the muscle-tendon mechanical properties? A review. Scandinavian Journal of Medicine & Science in Sports, 28(3), 794–806.

[2] Romero, Norma Beatriz (2013). [Handbook of Clinical Neurology] Pediatric Neurology Part III Volume 113 || Main steps of skeletal muscle development in the human. , (), 1299–1310.

[3] Maw, Jonathan; Wong, Kai Yuen; Gillespie, Patrick (2016). Hand anatomy. British Journal of Hospital Medicine

1.2 Objectives

Going back to the points mentioned before, it is all a matter of addition to the current situation. Moreover, the great ignorance that exists in this regard and the possibility of a forthcoming resurgence is what motivated us to start with the project. Eventually, leading to research and being able to take part in the growing sector of biomechanics. However, to be able to start a practical research, theory may be the first step, consequently we needed to investigate to then create. This article studies thoroughly the process of building a myoelectric prosthesis, and eventually leading to an external hand viable to use in different situations without regarding the distance between emissary and receptor.

On the one hand, the objectives related to the research and in which we will build this paper around are as follows:

I. Create a biomimetic hand that can replicate as much movements as an actual human one.
II. Enable the hand to move remotely without regarding the distance between the emissary and the receptor.
III. Reach the acceptance values to be a prosthesis.
IV. Create an affordable yet reliable biomechanical hand so it can be used around the world.

On the other hand, some objectives were set to provide some personal growths, these are:

I. Having a better knowledge of programming, 3D modelling, mathematics, and kinematics.
II. Develop our skills in the creation of a long-term project.
III. Learn and acquire experience the methodologies of the scientific method and laboratory research.
IV. Describe our findings to contribute in some way to the scientific community and be able to meet scientists.

1.3 Methodological question

Is it possible that 4 students can make a biomechanical hand which is able to move and be controlled by their own software from somewhere from the other side of the world?

1.4 Hypothesis

To achieve our objectives, we emanate from the following hypothesis:

I. It is possible to create a functional limb, controlled either with external or internal stimulus, in 9 months that can be used efficiently in most tasks.

II. There is a way to avoid latency problems during signals transmissions in which case human contact would feel like immediate.

We have based our hypothesis in the fact that creating a biomechanical hand has been done before. However, our hypothesis comes from our believe that if we use the set of tools, we dispose of this being: a 3D printer, some programming and modelling software's to name. Together with doing the research needed and some people giving us advice we believe we will be able to build from pretty much scratch a physic bionic hand which is mobile and controllable from as far as the MIT university.

1.5 Research methodology

To conduct our research, we will talk to some professionals with some experience in the area, this part is already settled as we have contacts in MIT and the experience of some relatives of ours that can help us too. As an example, we are working with a group in Ferrara, who have done a similar project and who have been giving us advice. As for the theoretical framework of the project, we will also need to consult several information sources such as: scientific articles, specialized books, reliable websites, interviews, etc.

Alternatively, the practical part relies on testing the prototypes in different areas to see mistakes that can be improved. We separate this research in two parts, on the one hand hardware and on the other hand software. Both play a critical role in the making of the biomechanical hand.

Firstly, hardware will be designed from scratch and to decide which ideas fits best the hand, testing is needed. For this task we will be testing each prototype, both hand and recognition system, with the same tasks being able to extract data that can later be compared to extract conclusions on what needs improvement and what does not. Testing may involve specific areas of the final design such as joints, servos, structural matters, wiring, coding, etc. and the characteristics tested are the following: strength, durability, effectiveness, comfortability/mobility, in addition to some others that are the most critical in the design of a biomechanical hand.

There will also be a good amount of coding involved in the project, to connect the hand to the movement recognition system some go-between needs to be established, this one needs some characteristics suitable for the job. All this coding is accessible by everyone and is stored in our GitHub Page[4]. As in the hardware, testing may be made to extract conclusions through the data recorded. During this process we will consult any doubt to our mentors at the MIT and around the globe, who will guide us so that we find the best solutions for our issues and who will help us understand our mistakes.

[4] https://github.com/Shaking-Hands-Overseas

Once we have the needed knowledge and the best approach to each problem we encounter, we will proceed to design and build the final prototype of the limb and hand movement detection system. During this process we want to discard each non-viable idea and end up with the most efficient design possible. When we have our final prototype ready for testing, we will try and connect with MIT for our final test in a close loop, so they would have a hand recognition system and a hand that we would move and vice versa. which is to have a verdict of the correct functioning of every part of software and hardware we designed and prepare it for the day we shake hands from thousands of miles away.

2. Theoretical Framework

2.1 General anatomy

The human body is already a machine of itself which has had some million years to perfect itself. So, we will be trying to recreate in a few months a part of this incredible engineering masterpiece that are our bodies. To complete this task, we will do as we do to learn, imitate. For it we will first need to investigate how the motor functions work for our selected body part, the forearm. In our case we will be focusing on the joints of it. That is because, the structure of our 3D contraption can hold on its own, making the job of the bones and ligaments, so we do not need use of them nor the muscles, since we also dispose of our Arduino motors that serve for that purpose and do not need to be placed in specific positions nor working in a set way. On the other hand, for the sake of keeping the shape of our human forehand, because in fact if we wanted to, we could make a different type of limb having the same purpose of our regular forearms, we must place the joints in kind of their respective place of the hand. Furthermore, the joints also help us figure out how is it possible for the different bones and cartilages to move which is a point we need to mimic as we work through strings, making it similar.

2.1.1 Hand joints

(A joint or articulation is a structure in vertebrate organism at which two bones join.)
To picture a better idea of the hand joints we will first need to know that there are 27 bones in it, this can be a shock as they are quite a lot for the little size of this body part. The great density of this bones is in the wrist of the hand that is why our hand is not completely stiff without it being a flesh and able to do bend in many ways, like being able to touch our thumb with our little finger or squeezing it with no problem.
If we would have to put our hands through x-ray, we would see that our fingers “extend more than what they seem.” What it means is that instead of the three bones that form our fingers, that we can tell, there is a fourth one which forms the palm of the hand, this is

except for the thumb which has the "extension" of two bones. With the picture following picture you will be able to see it better.

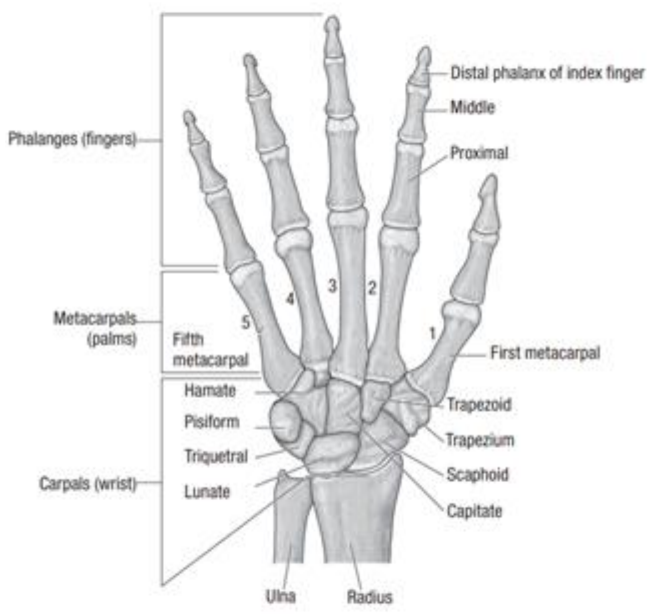


Figure 1 - Anatomic representation of the bones of the human hand including Radius and Ulna.

Both the metacarpophalangeal joints, the joints of the bones of our palm, and the interphalangeal joints, the joints of our fingers, are synovial, these are the joints that unite them both. Synovial joints are a type of joint which joins long bones through a joint capsule to get a greater mobility letting them do the following movements: extension and rotation. Instead, the joints that join the metacarpals with the carpals are condylar, which allow flexion, extension, abduction, and adduction movements. Interphalangeal joints in the other hand, the joints that join the phalanges, are of the hinge type permitting only flexion and extension movements. The joints although we have described them with having many different movements, when set on trial by ourselves, cannot move that much. This is due to the collateral ligaments which limit the joints movements, not letting the fingers fall apart.

2.1.2 Wrist joints

The wrist is another complex part of the body and although it is mentioned before that the carpals are part of the hand; it is half true. This is because we could also argue that they are part of the wrist, the point, is that the wrist is inside of the hand. It depends on the source you look for, for what we can tell. There are 8 carpal bones in the wrist with multi articulated joints. The 8 bones form two rows that form the radiocarpal, midcarpal and carpometacarpal joints, with intercarpal joints lying between the individual bones. The carpal joints are supported by ligaments and there is little movement of this bones. These ligaments are of importance as the disruption of one of them can result in wrist instability. What we can take from this is that each individual bone in our wrist/palm is movable with a network between them helping to hold them and the hand up. But in our case, we will not be needing much reference to it except for the trapezium, the carpal next to the thumb, since will be featuring it. Its function is quite simple, it creates most of the movement on the extremity, it holds up the thumb and allows more movement, flexion, extension, abduction, adduction and circumduction.

- Flexion: movement that decreases the angle between two body parts.
- Extension: movement that increases the angle between two body parts.
- Abduction: motion of the limb that pulls away from the midline of the body.
- Adduction: motion of the limb that pulls towards the midline of the body.
- Circumduction: motion of a circle

On the other hand, the index and middle fingers joint have relative stability, which means their move span is shorter than that of the thumb joint. Additionally, the fourth finger joint is more mobile than the two fingers mentioned before, furthermore, the fifth joint allows cupping of the hand.

2.1.3 Elbow joints

The elbow is a trochleogingylomoid joint formed by three main joints. It is called a trochleogingylomoid joint as it can flex and extend (ginglymoid) as well as do the pronation-supination, supination is the movement that allows the palm of the hand to be facing upwards; pronation is the rotation of the forearm so that the palm is facing downwards, movements (trochoid).

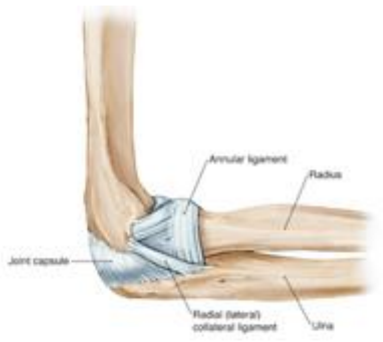


Figure 2 – Lateral representation of joining bones within the elbow and its joint.

The three joints forming it are:

- The ulnohumeral joint: it joins the humeral and ulna bones articulation with a hinge joint, a joint in which the articular surfaces, in this case of the humerus and the ulna, are shaped in a manner to permit motion in only one place.
- The radiocapitellar joint: it joins the humeral and radial bones articulation.
- The proximal radioulnar joint: it joins the radial head, the radial part nearest to the elbow, with the radial notch, a small depression along the later surface of the ulna.

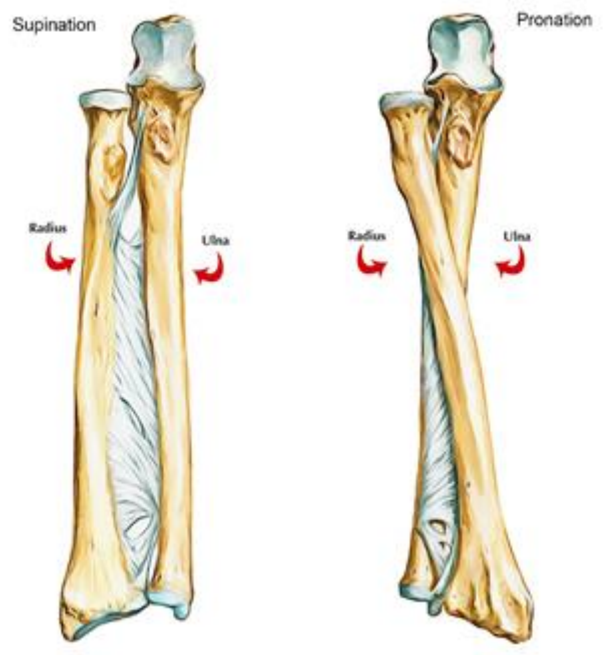


Figure 3- Positioning of radius and Ulna when hand palm facing upwards (left), and when hand palm facing internally.

It joins the humerus radius and ulna bones of the forearm with the humerus of the upper arm; it also joins the muscles as the biceps and triceps with the extensor muscles in the forearm; there are also tendons attached to it with even neurovascular structures going through. The elbow joint allows most of the movement of not just the wrist but the entire arm including the shoulder. As it is the choke point of the arm acting as an intersection and a stabilizer for the whole arm.

2.2 The muscular system

Human body is composed by various interactions of complex systems that enables life as we know it. Focusing on the muscular system, we refer it as an organ system formed from skeletal, smooth (non-striated), and cardiac muscles. It allows movement, maintains posture, and circulates blood throughout the body. It is important to bear in mind that although some muscles, such as the heart muscle, are entirely autonomous, the nervous system controls the muscular systems of vertebrates with electrical impulses, it is for this reason that the nervous system should be included in the theoretical part of this research.

Skeletal muscle is one of three major muscle types, the others being cardiac muscle and smooth muscle. It is a form of striated muscle tissue which is under the voluntary control of the somatic nervous system. (Birbrair, Alexander et al., 2013-03-21) Most skeletal muscles are attached to bones by bundles of collagen fibers known as tendons and receive inputs trough nerves.

2.2.1 Skeletal muscle structure

Skeletal muscle is one of the three muscle types we find in our body, it is known to be the principal muscle type that provides movement to the body. Skeletal muscle is an organ that comprises of different coordinated tissues. These tissues incorporate the skeletal muscle filaments, veins, nerve strands, and connective tissue. Each skeletal muscle has three layers of connective tissue that wall it in, give construction to the muscle, and compartmentalize the muscle filaments inside the muscle (Figure 10.2.1).

Each muscle is enclosed by a sheath of thick, unpredictable connective tissue called the epimysium, which permits a muscle to move capably while keeping up with its primary structure. The second being the epimysium, whose function is to isolate muscle from different tissues and organs nearby, permitting the muscle fibers to move freely with the minimum friction. Inside each skeletal muscle, muscle filaments are coordinated into groups, called fascicles, encircled by an inner layer of connective tissue called the perimysium. This fascicular association is normal in muscles of the appendages; it permits the sensory system to trigger a particular development of a muscle by actuating a subset of muscle filaments inside a fascicle of the muscle. Inside every fascicle, each muscle fiber is encased in a dainty connective tissue layer of collagen and reticular filaments called the endomysium. The endomysium encompasses the extracellular matrix of the cells and assumes a part in moving power created by the muscle filaments to the ligaments.

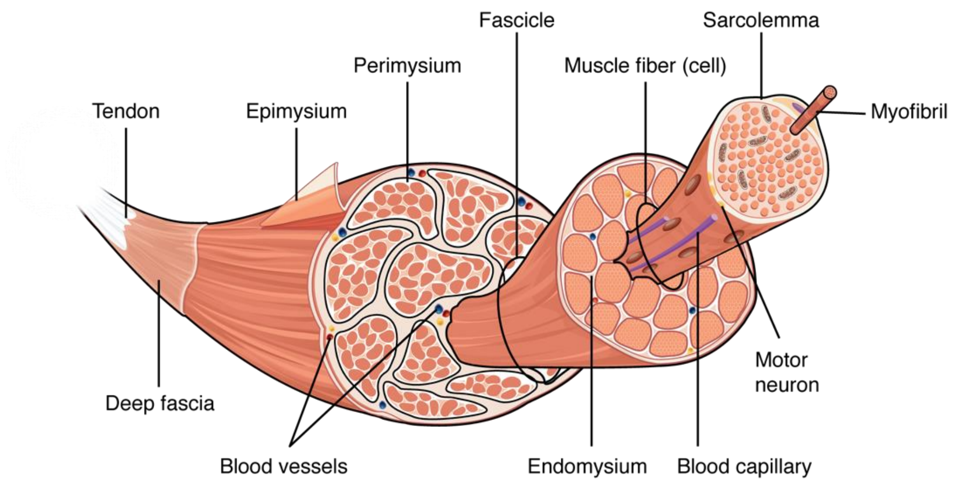


Figure 4 - Connective Tissue Layers froming the basic structure: The fascicles, groups of muscle fibers, covered by the perimysium. Muscle fibers are covered by the endomysium.

In skeletal muscles that work with ligaments to pull on bones, the collagen in the three connective tissue layers interlaces with the collagen of a ligament. At the opposite finish of the ligament, it wires with the periosteum covering the bone. The pressure made by compression of the muscle strands is then moved however, the connective tissue layers to the ligament, and afterward to the periosteum to pull on the bone for development of the skeleton. In different spots, the mysia may meld with an expansive, ligament like sheet called an aponeurosis, or to belt, the connective tissue among skin and bones. The wide sheet of connective tissue in the lower back that the latissimus dorsi muscles (the "lats") combine into is an illustration of an aponeurosis.

Each skeletal muscle is additionally luxuriously provided by veins for sustenance, oxygen conveyance, and waste expulsion. Also, every muscle fiber in a skeletal muscle is provided by the axon part of a substantial engine neuron, which flags the fiber to contract.

2.2.2 Skeletal muscle fibres

Since skeletal muscle cells are long, round and hollow, they are usually alluded to as muscle filaments (or myofibers). Skeletal muscle strands can be larger in contrast with different cells, with lengths varying from 100 µm to 30 cm, for example in the Sartorius muscle of the upper leg. Having numerous cores takes into consideration creation of a lot of proteins and compounds required for keeping up with typical capacity of these protein base cells. Moreover, skeletal muscle strands additionally contain cell organelles found in different cells, for example, mitochondria and endoplasmic reticulum. However, a portion of these constructions are well versed in muscle filaments. The smooth endoplasmic reticulum, called the sarcoplasmic reticulum (SR), stores, delivers, and recovers calcium particles (Ca++)[1].

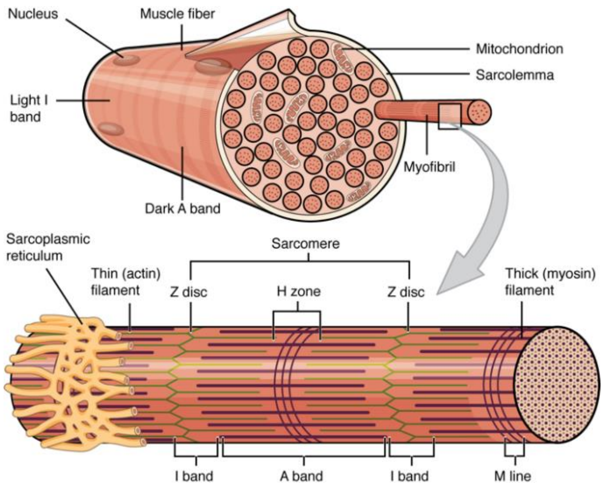


Figure 5 - Muscle Fiber: A skeletal muscle fiber is surrounded by the sarcolemma, which contains sarcoplasm, the cytoplasm of muscle cells. A muscle fiber is composed of many myofibrils.

The plasma film of muscle filaments is known as the sarcolemma and the cytoplasm is alluded to as sarcoplasm (Figure 5). Inside a muscle fiber, proteins are coordinated into structures considered myofibrils that run the length of the cell and contain sarcomeres associated in series. Since myofibrils are just around 1.2 µm in breadth, hundreds to thousands (each with substantial number of sarcomeres) can be found inside each muscle fiber. The sarcomere is the littlest practical unit of a skeletal muscle fiber and is an exceptionally coordinated course of action of contractile, administrative, and underlying proteins. It is the shortening of these singular sarcomeres that lead to the constriction of individual skeletal muscle filaments (and at last the entire muscle).

2.2.3 Muscle fibres development

To start with, muscle tissues are derived from the mesodermal layer of embryonic germ cells in a process known as myogenesis. There are three types of muscle, skeletal or striated, cardiac, and smooth. Muscle action can be classified as being either voluntary or involuntary. Cardiac and smooth muscles contract without conscious thought and are termed involuntary, whereas the skeletal muscles contract upon command. (Mackenzie, Colin, 1918) Skeletal muscles in turn can be divided into fast and slow twitch fibers[2].

Bearing all these in mind, Satellite cells of the skeletal muscle are located between the basement membrane and the sarcolemma (cell membrane) of human muscle fibers. They can differentiate and fuse, multiply current muscle fibers and form new muscle fibers. These cells represent the person's oldest recognized cells and are involved in the regular growth of muscles and recovery after injury or illness.

The cells of the inner cell mass (embryonic cells), recognized as human embryonic stem cells (hESCs), are distinguished to form four structures: the amniotic membrane, the yolk sac, the allantois, and the embryo itself. Human embryonic stem cells are pluripotent. That is, they can be distinguished by any type of mobile phone that exists in adult humans, and any type of intermediate parent phone that later becomes an adult telephone line. HESC is also immortal. Unless the current process is differentiation or cellular senescence (cellular senescence), they can divide and develop indefinitely in a series.
The first differentiation of hESC, which forms the embryo itself, takes place in three phone types, ectoderm, mesoderm, and endoderm, which are recognized as germ layers. The ectoderm then alters the pores and skin (including hair and nails), mucous membranes, and affected systems. The mesoderm represents the skeleton and muscles, the heart and circulatory system of the coronary arteries, the urinary and reproductive systems, and the internal connective tissue of the body. Endoderm forms the gastrointestinal tract (stomach and intestines), airways, and endocrine devices (liver and endocrine glands).

The process of myogenesis: myogenesis is the formation of muscle tissue, especially during embryogenesis. Myoblasts are made up of myoblasts fused into polynuclear fibers called myotubes. In early embryonic development, these myoblasts proliferate if there is sufficient fibroblast growth factor (FGF). When FGF is depleted, myoblasts cease to divide and secrete fibronectin into the extracellular matrix. In the second stage, the myoblasts in the tube are aligned. Phase 0.33 is the correct merger of the mobile phones themselves. At this stage, calcium ions are essential for development. Muscle cell beautification elements (MEFs) promote myogenesis. Serum response (SRF) plays a vital role during myogenesis. This is required for the expression of the alpha actin striped gene. Skeletal α-actin expression is further regulated by androgen receptors, and its potential steroids can alter myogenesis[5] .

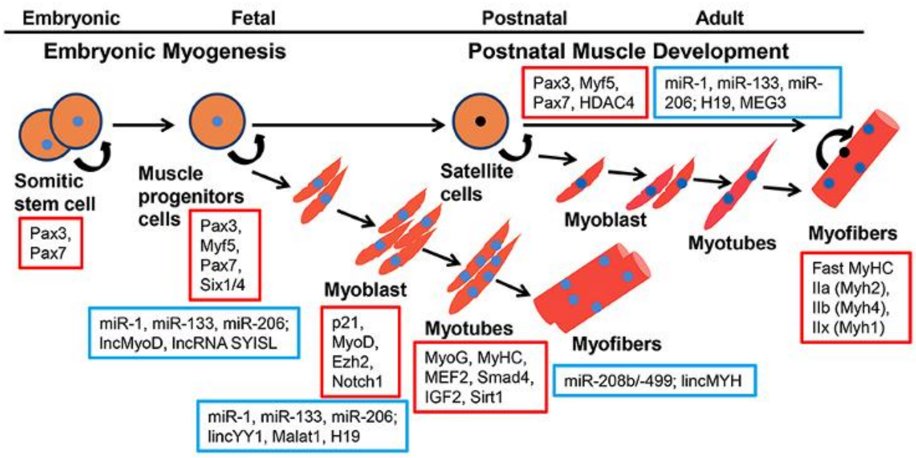


Figure 6- Muscle cells development: A somatic stem cell of an embryo undergoing differentiation and fusion to get to myofibers, during the embryonic, foetal, postnatal and adult phase.

Myoblasts are a type of embryonic differentiation that forms muscle cells. Skeletal muscle fibres are formed when myoblasts fuse, so muscle fibres have multiple nuclei. Myoblast fusion is specific for skeletal muscles (e.g., triceps brachii). Myoblasts can differentiate, these cells remain adjacent to the muscle fibers located between the sarcolemma and the endomysium, the connective tissue that divides the muscle bundle into human fibers. Satellite cells can differentiate and fuse to augment existing muscle fibers and form new muscle fibers. In healthy muscles, most satellite tvs for computer cells are inactive. It does

[5] *Parvini, F., & Shahabi, C. (2007). An algorithmic approach for static and dynamic gesture recognition utilising mechanical and biomechanical characteristics. International Journal of Bioinformatics Research and Applications, 3(1), 4.*

not distinguish or support the mobile sector. In response to mechanical stress, satellite cells emerge in an activated state and initially proliferate as a skeleton myogenic cell.

In conclusion, embryonic myogenesis is the process in which somatic stem cells undergo a process of differentiation and fusion in which they finalize by gaining characteristics and properties that make movement as we know it.

2.2.4 Functions of the skeletal muscle

The main function of skeletal muscle occurs through the unique binding process of excitation-contraction. This process takes place with the union of muscle, bones, nerves and veins. The main functions are as follows:

- Skeletal muscle also provides structural support and helps maintain posture. While the muscles are attached to the tendons of the bone, the contraction of the muscle leads to the movement of the bone, allowing it to perform certain movements. To replicate a human hand, this function needs to be as close as we can get. Mobility and flexibility are key elements to provide a both functional and useful biomechanical hand.

Focusing on the hand-arm movements we can observe that during an eccentric contraction of the biceps muscle, the elbow starts the movement while bent and then straightens as the hand moves away from the shoulder. During an eccentric contraction of the triceps muscle, the elbow starts the movement straight and then bends as the hand moves towards the shoulder. There are both physical and chemical requirements to get to move a muscle, but in our case, the chemical requirements are more about electric current and circuits[6] [7] [8].

[6] *Martin, Silvia; Sanchez, Eugenia (2013). Anatomy and Biomechanics of the Elbow Joint. Seminars in Musculoskeletal Radiology*
[7] *Karbach, Lauren E.; Elfar, John (2017). Elbow Instability: Anatomy, Biomechanics, Diagnostic Maneuvers, and Testing. The Journal of Hand Surgery*
[8] J.N.A.L. Leijnse; J.J. Kalker (1995). *A two-dimensional kinematic model of the lumbrical in the human finger., 28(3), 237–249.*

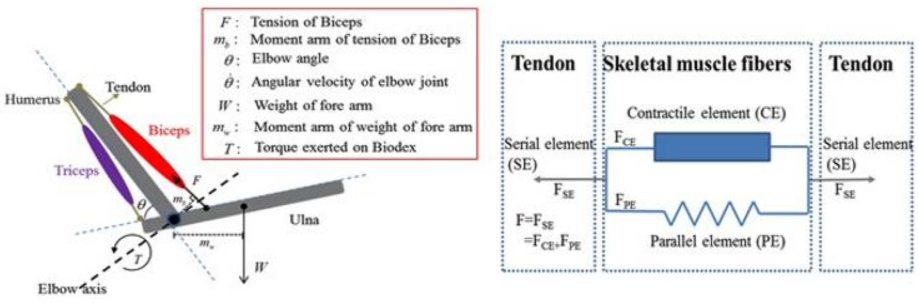


Figure 7- Biomechanical model for elbow movements: A model that shows the physical requirements to replicate de elbow movement, to understand how bones and muscle connections work.

-Skeletal muscle also functions as a source of amino acids that various organs in the body can use to synthesize organ-specific proteins.

-Skeletal muscle also plays a significant role in maintaining thermostatics and serves as an energy source during famine.

It is important to bear in mind how does muscular-nervous interaction works. Motor control is a field of study relatively young; it may be defined as a scientific area studying the way the central nervous system (CNS) produces voluntary and purposeful movements in its interaction with the rest of the body and its environment. The understanding of it helped us understand and gave us the knowledge we needed to recreate the whole system[9].

[9] Check *Theories and control models and motor learning: Clinical applications in neurorehabilitation. Neurología (English Edition), 30(1), 32–41.*doi:10.1016 for more information

2.3 Materials

We have used a huge variety of different materials for a lot of purposes. We used electronics like motors, wires, Arduinos… We also used plastics for building the various parts of the hand and a large variety of strings. Also, in order to make the glove to control the hand we used a glove, wires, and flex sensors.

2.3.1 Materials for the hand carcass:

As for the hand we knew what we wanted to use, 3d printing to create all the models of the hand and parts for the glove, which we did. Through the process, we discarded making wood models because that would take us a lot of time, and metal models because it would have been too expensive for us as well as, we would need advanced machinery which we did not dispose of. Although it may look like a downside, with the 3d printer, you can print everything you want. The problem comes when choosing the material, there are a large variety of materials for printing, all of them are plastics but with different characteristics, there are plastics that are stronger than others like ABS, others that are elastic like PETG or more standard like PLA that is a little flexible and very strong. We decided to use PLA as it is the most standard.

2.3.1.1 Sustainability

This project was aimed to be not only cheap, but carbon neutral and eco-friendly. The plastic used (PLA) has been chosen because it is biodegradable via hydrolysis, thermal decomposition and photodegradation. Also, this material is obtained from natural sources like corn and as stated in "Journal of Polymers and the Environment" by Kerry Kirwan, PLA has a 75% reduction in carbon footprint versus traditional plastics. Which makes PLA the best choice environmentally speaking. A part of PLA we did not use any material with environmental impact.

2.3.2 Strings

To recreate the opening and closing of the hand we used strings. On the first prototype chose to use a stronger plastic string, it required to be strong otherwise it may break, when we finished the prototype, we saw that the string had iron in it which caused a huge friction within the hand structure damaging it, so we finally decided to discard this type. On the second and third prototype we used fishing strings instead, a huge downside was that they could brake much easily, in fact, we broke some on the 3rd and 2nd prototype which was not the case for our 1st one. The reason the strings broke was because in our 2nd prototype me made it so that the hand closed in a different way than in our 1st one, so the strings required to be stronger, and for our 3rd one the strings unleashed because of some more modifications to the plastic cover of the hand. The bad thing about the strings breaking is that replacing them is a tedious task.

2.3.3 Motherboard

The third type of material that we used, and we are using on our prototypes is electronics. We used diverse types of electronics for different purposes. First, we have the motherboard, then DC motors and servos, also wires and batteries.
When choosing the motherboard, we had to look for the different features between them.

Figure 8 - Image of an Arduino taken from the Arduino website.

Source:https://cdn.shopify.com/s/files/1/0438/4735/2471/products/A000066_03.front_763x573.jpg?v=1629815860

We were thinking on using a Raspberry Pi or an Arduino because they are not very difficult to use and we know how to program them, also there are a lot of sensors compatibles with them.

Figure 9 Image of a Raspberry Pi taken from Wikipedia.

Source:https://es.wikipedia.org/wiki/Raspberry_Pi#/media/Archivo:Raspberry_Pi_4_Model_B_-_Side.jpg

There are different things to consider while choosing the motherboard, the power source is a very important one. We needed that the mother board had an external source of power because we needed to use a lot of servos, and we had to make sure that the board had enough power to run all of them and herself. Another feature to consider is the memory of the motherboard. We did not know how large the memory space of the code would be so we had to make sure that it was large enough so that the memory of the board could late hold it.

Probably, the thing that made us chose the Arduino was the code. For the programming language on the Raspberry Pi, you must use python while on the Arduino you must use C++ (unless you change it, but C++ is the default language). Also, we had used Arduino before, so we knew how to program sensors using C++, but we have never programmed a sensor with python, so we ended choosing the Arduino. Another factor was that we knew which sensors were compatible with the Arduino.

2.3.4 Servos

Regarding the fingers, we used servo motors with a 3d printed peace on the top of the servo which holds the string still. This is a cylindric piece of plastic with a canal on the side that is where the string goes through.

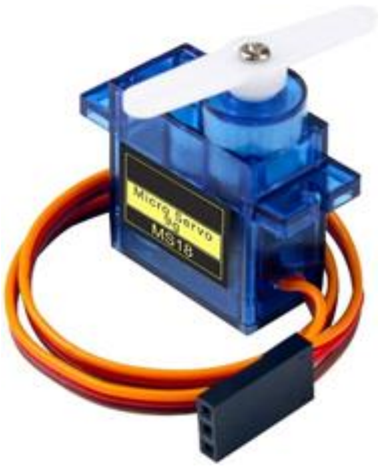


Figure 10- Image of a Micro Servo 9g MS 18 taken from Amazon (It is the same model as the Micro servo SG 90 and it have the same features). This motor is programmed writing angles and it has a maximum rotation of 180º.

Source: https://m.media-amazon.com/images/I/61-JLAIqIIS._AC_SL1283_.jpg

When we first started the project, the servos that held this piece were the basic model of servo, the sg90. These servos could not handle the force needed so we had to change them for others that had the capability. We changed them on the second prototype because this prototype had the rubber bands on the back to maintain the fingers straight, so the servo just needed to turn to close the fingers and return to the initial position so that the fingers would return to their original position with the help of the rubber bands force. The servos we found capable for the task and the ones we finally used were the MG996R, which we used 6 of, this servo can handle a lot of force, but they are bigger and heavier so we needed to change the design and so we did. When we changed the small servos for the bigger ones, problems appeared, we needed more energy because just three of them were working. They could handle the force, but they needed more power. So, we added an external source of energy witch it was a 12V battery and an electronic device that regulates the power of the battery to the servos.

Figure 11- Image of a DC converter modified from amazon.

Source:
https://www.amazon.es/dp/B08S376RHL/ref=cm_sw_em_r_mt_dp_J33827YCC4BQ4BA235X4?_encoding=UTF8&psc=1

2.3.5 Cables

There are three different types of cables, the difference between them can be found on the ends, the cables can end with a pin and a hole(M-F), a hole and a hole(F-F) or pin and pin(M-M). As for the type, we used the basic Arduino cable, for the most part we used M-F type, but sometimes we needed to the F-F type ends.

Figure12 - Image of cables F-F taken from nelbren.

Source: https://nelbren.com/assets/images/posts/cables_hembra_hembra.png

2.3.6 Power source

We had trouble with the servos because they could not resist the force of the rubber bands and we needed to change them, the power that the computer gave to the Arduino board was not enough for all the five big servos. That is because the computer USB port it gives a maximum of 5V, so we needed to connect an external source of power, in our case, we used a 9V battery with two cables that connect to the Arduino, this enables all the servos to move. The five servos make the battery wear out very fast because they require more power. This did not happen with a 12V battery that is the one we ended up using.

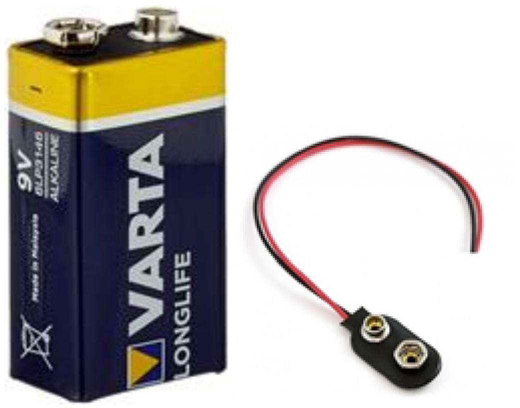


Figure 13 - Image of a 9V battery connector taken from BricoGeek.com.

Source: https://tienda.bricogeek.com/2174-thickbox_default/cable-para-pila-de-9v.jpg

Figure14 - Image of a 12V battery

Source: https://www.amazon.com/-/es/Mighty-Max-Battery-ML15-12-Producto/dp/B00K8V2FUE/ref=sr_1_10?_encoding=UTF8&c=ts&keywords=12v%2Bbatteries&qid=1637007713&s=hpc&sr=1-10&ts_id=389574011&th=1

MG996R Digital servo	· Operating voltage: 4.8-7.2V · Operating speed: 0.17sec/60 degrees · Torque: 14kg-cm · Degree: 180º
Arduino UNO	· Operating voltage: 5V · Input voltage: 7-12V · Digital I/O pins: 14 · Analog pins: 6 · DC Current per I/O Pin: 20 mA · DC Current for 3.3V Pin: 50 mA
DC converter	· Input voltage: 8-22V · Output voltage: 1-15V · Output current: 3A (max)
Battery	· Output voltage: 12V

2.4 Coding Theory

2.4.1 Introduction

For a project of this kind, we need to understand the protocols of the web and how the information is saved, processed, and transmitted through the internet. This part will review the different options which are available for this kind of communication and their usage and benefits. It also will provide an overview of the existing hardware. Its functions and an approach to our custom-built source code.

2.4.2 The Frontend

2.4.2.1 What is the Frontend?

The Frontend is a website or GUI (Graphical User Interface), which is a way to share and access information using a web browser. A Frontend it is the only opened to the public part of a server and used to communicate with the "invisible" part, which is the backend. It is what the user can see and talk with, while background processes interact with the API requesting and sending data whenever the user wants to see or create something. Consequently, without the API, the Frontend is static, and it would only show the page itself without further functionality, but it would still be accessible, taking into consideration that the information may not show appropriately. Also, a frontend lets the user interact with the API in a more "humanistic" way, by clicking buttons and slicers, and it is mainly used to provide the user with interactive information, which changes depending on the user for a more personalized experience.

2.4.2.2 Where is it built on?

The standard languages used in a Frontend are JavaScript for the actions as clicking buttons or changing the page in some way, HTML for the page itself and CSS or SCSS for the page style. These are used together to obtain improved control over the page. As an example, we could create a dropdown menu in HTML, and with a style we determine with our CSS code, which would show up depending on the status of a variable in our JavaScript code.

So, HTML and CSS are used for the page without functionality and JavaScript to add this functionality to our page.

2.4.2.3 Frameworks

A framework is used to better manage routing and security, which is a code library with plenty of functionalities already coded in and ready to use, they are most common for their reliability and simplicity, consequently improving workflow and debugging. There are lots of frameworks available to the public, and most of them are supported by big companies such as Facebook or Google and with enormous communities backing them. Each Framework is specialized in a certain feature or philosophy, that distinguishes it from others, what makes choosing a hard and farsighted activity. As an example of this philosophy, a Frontend Framework can be a single-page application (SPA) or multi-page application (MPA):

- A single-page application is one page with only one distribution, but it also can show different content as the information shown can vary depending on what the user wants to see or can see. A good example of this Is Gmail, the email web from Google, where the distribution does not change, but the content does, depending on the emails you received or the section you are in. This discharges the server from sending whole pages each time a request comes in and instead, sending only what it is needed, improving performance and latency.

- A multi-page application is organized in routes, which contain a different html page with different code. Contrary to a Single-page application, each time a request comes in, the server sends a new entire html page, instead of only the content that may change. This affects performance, but if a webpage does not use the same headers and footers in all pages and needs to be static or its more efficient using multiple pages, this Is the recommended way to go. An example of this is Amazon and online shops, as multi-page applications tend to be safer, easier and scalable.

The most used options of Frontend frameworks following the *2021 Stack Overflow survey* are[10]:

- **Angular**: Which provides more predictability in the coding but less flexibility. It is based on “components,” which are a combination of HTML, CSS and JavaScript code which run together to perform a concrete action or functionality. An example of this would be a component which requests things to an API and contains a certain functionality or simply the content of a page. Angular is a single-page application framework, so its pages are uploaded dynamically, with static parts and components which load whenever the user requests their use.

- **React** on the other hand, is more minimalistic and flexible than Angular. But it requires more decision-making because of its minimalistic philosophy, as you must add plugins and other smaller frameworks when you want to use routing or animations. This part of decision-making may be overwhelming if you have little or no experience in web applications, what makes a much harder experience.

- **VueJS** Is similar to angular, but with a more simplistic approach. Instead of using a folder for each component, they are integrated into a single file, and it also has independent dependencies and plugins, but they are not as powerful as the ones in angular.

[10] https://insights.stackoverflow.com/survey/2021

2.4.3 The Backend

2.4.3.1 What is the backend?

The backend is the part of a server where all the processing, authentication and storage is made. This part is not directly accessible by the user, although it can be accessed if the host wants to, and it handles the requests made by the Frontend or by a program dependent on a server.
A backend may have lots of parts, but the main ones are the API and the database, which we will cover in this part.

2.4.4 The API

2.4.4.1 What is an API?

An API (or Application Program Interface) is the core of all communication through the internet and provides access to the information needed stored in the backend and returns it to the Frontend. It is used in every cloud infrastructure because it provides control over all the key parts of a web server. As an example, when you search for "cats" in google, the request is sent to an API, which sends a query to the database, which returns to the API the most common results for "cats", and the API returns it to you.

2.4.4.2 Types of APIs

There are two most used types of APIs used in the world. Rest and GraphQL.

- Rest API:

  Rest APIs are the APIs which have been around since Tim Berners-Lee created the world wide web and are the ones which are most used in the present, as they are the current architecture of the World Wide Web. Rest APIs have a simple and scalable approach, where you must decide how you want your API to work and memory allocation. Also, they are safe and reliable and with intuitive working, but the downside is that is static and with little flexibility in terms of data types because if you request something, you will receive what the coder coded in as the response to your request, not what you wanted. So, you might receive lots of useless data when you only need a specific argument.

- GraphQL:

GraphQL APIs are based on sending only the data requested and avoiding sending useless data which the user does not need, also sending it back in the data type the user wants. These have major performance and simplicity boosts in large projects and webservers, as they could save bandwidth by only sending back the needed data in one connection only, but for smaller projects they tend to be overkill, as they do not benefit that much of the use of one and RestAPIs are simpler and more reliable for them.

2.4.4.3 How does communication between a User and a Server Work?

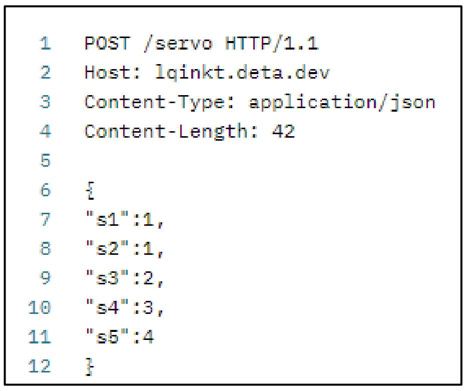
An API is always listening to requests to answer and are intended to be used as a distributor of information and credentials, as they are the ones who verify users and provide information to the user such as profile pictures, names, descriptions, … A good example of the work an API does Is when logging into a social media page: First, the user device will send a request using the device cookie or authentication token provided by the API previously and ask for the last stories and pictures, the ones you follow posted. This will trigger an internal process in the API where it will search in its database for your followers and recommendations, go to each profile you follow and see its last activity and finally send it to you in a data type your device can handle. Then, your device will process the received data and store it in your device memory, so when that information is needed, your device will output that data into your screen, and it will repeat the process after some seconds to verify and keep everything updated. This is a simplification of the process as many parts are involved, but API communication looks like the latter.

Rest API Communication: For any kind of request to a Rest API, the HTTP methods are used as a standard for all communications on the internet, which are described as following in the *Mozilla Web Documentation* [18]:

GET: The GET method requests a representation of the specified resource. Requests using GET should only retrieve data.
HEAD: The HEAD method asks for a response identical to a GET request, but without the response body.
POST: The POST method submits an entity to the specified resource, often causing a change in state or side effects on the server.
PUT: The PUT method replaces all current representations of the target resource with the request payload.
DELETE: The DELETE method deletes the specified resource.
CONNECT: The CONNECT method establishes a tunnel to the server identified by the target resource.
OPTIONS: The OPTIONS method describes the communication options for the target resource.
TRACE: The TRACE method performs a message loop-back test along the path to the target resource.
PATCH: The PATCH method applies partial modifications to a resource.

Figure 15 - Definition of each existent HTTP Method

These methods are used as in the following example, where a POST method is used to request some data to a server:

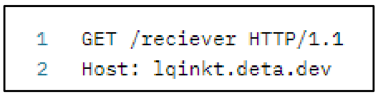


```
POST /servo HTTP/1.1
Host: lqinkt.deta.dev
Content-Type: application/json
Content-Length: 42

{
"s1":1,
"s2":1,
"s3":2,
"s4":3,
"s5":4
}
```

Figure 16 - POST request written in HTML

Source: Postman

In these kinds of requests, we start with the headers, where the information about the communication is stored. The headers start by saying the method we want to use, in this case POST, which we will use to send some data to a server. Later we specify the server route, which is where the part of code we want to use to receive this data is, in this case "/servo". Finally, we end the first part with the version of HTTP (Hyper transfer protocol) we are using which will determine the type of response we want back and request we are sending. Just before the actual information we have the server DNS where we want to send the request, this will be transformed into an IP address by a DNS server provider. Now we have the information we are sending. We start with the type in which this is stored, in this case a JSON file, later the length of the content we are sending and finally the information. Another HTTP method commonly used is the GET request, which is used to request some data to the API. The headers of a GET request are much simpler than a POST one, as they only contain the HTTP method, route, HTTP version and the server DNS. As the data we will receive is in a fixed format.

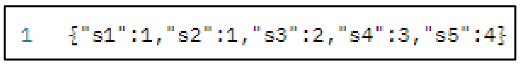


```
GET /reciever HTTP/1.1
Host: lqinkt.deta.dev
```

Figure 17 - GET request written in HTTP

Source: Postman

Then the user sending the request will receive back a JSON file with the information requested which would look like the following.



```
{"s1":1,"s2":1,"s3":2,"s4":3,"s5":4}
```

Figure 18 - Rest Server response to a GET request

Source: Postman

In this case, we are receiving back a simple JSON with some information, but we might receive enormous files with lots of variables.

<u>GraphQL Communication:</u>

A GraphQL communication is much simpler than a RestAPI one, as the user requesting data sends a JSON-encoded string containing the requested information and as a form, the API fills a JSON with the requested information and sends it back. This communication is clearly explained in the *GraphQL documentation*[11]:

> GET REQUEST
>
> When receiving an HTTP GET request, the GraphQL query should be specified in the "query" query string. For example, if we wanted to execute the following GraphQL query:
>
> ```
> {
>   me {
>     name
>   }
> }
> ```
>
> This request could be sent via an HTTP GET like so:
>
> ```
> http://myapi/graphql?query={me{name}}
> ```
>
> Query variables can be sent as a JSON-encoded string in an additional query parameter called `variables`. If the query contains several named operations, an `operationName` query parameter can be used to control which one should be executed.

[11] https://graphql.org/learn/serving-over-http/

The response, as said, comes in a JSON file, where the data comes together with the errors that might have happened. The following example from the *GraphQL documentation* shows clearly this JSON file*:*

```
{
  "data": {...},
  "errors": [...]
}
```

Figure 19 - Response from a GraphQL Server to a GET request

Source: https://graphql.org/learn/serving-over-http/

In a response from a server, it is also included the status code, which is an indicator of how the connection went. These status codes indicate whether there is an issue or not and where this issue may come from. The two most common ones are described as follows in the *Mozilla Web Documentation*[12]:

200 OK

The request succeeded. The result meaning of "success" depends on the HTTP method:

- GET: The resource has been fetched and transmitted in the message body.
- HEAD: The representation headers are included in the response without any message body.
- PUT or POST: The resource describing the result of the action is transmitted in the message body.
- TRACE: The message body contains the request message as received by the server.

404 Not Found

- The server can not find the requested resource. In the browser, this means the URL is not recognized. In an API, this can also mean that the endpoint is valid but the resource itself does not exist. Servers may also send this response instead of 403 Forbidden to hide the existence of a resource from an unauthorized client. This response code is probably the most well known due to its frequent occurrence on the web.

Figure 20 - Definition of the two most common Status Codes

Source: https://developer.mozilla.org/en-US/docs/Web/HTTP/Status

[12] https://developer.mozilla.org/en-US/docs/Web/HTTP/Methods

2.4.4.4 Backend Frameworks

Frameworks are libraries of code which enable the user to create an API. These are supported by foundations or companies, and they facilitate the creation of new APIs.

There are lots of frameworks in existence for an API, but the ones more used following the *2021 Stack Overflow Survey*[13] are:

- **Laravel**: Written in PHP.
- **Flask**: Written in Python.
- **Django**: Written in Python.
- **Springboot**: Written in Java.
- **ExpressJS**: Written in JavaScript.
- **ASP.NET Core:** Made by Microsoft and written in C#.

All these Frameworks have lots of functionalities and peculiarities, consequently, it usually depends on the user to choose according to his preferences.

2.4.4.5 Authentication methods

To distinguish users and verify each person is who they claim they are, APIs usually use one of these two different methods: Session ID or Token Authentication.

- Session id or commonly known as cookies, are strings or small data packages which are sent from the server when logging into a website and are stored into the user's browser and act as a key when sending requests to a server. Consequently, whenever the user wants to retrieve some data from a server, it will send this cookie to the server, so that the server knows who is speaking with. If it is the first time we have spoken with that server, a new cookie will be created. This cookie will be stored both in the server and client.

- Token Authentication works like a session id authentication, but instead of creating a cookie it creates a Json Web Token (JWT), which is created with a private key in

[13] https://insights.stackoverflow.com/survey/2021

the server. Whenever the user wants to talk with the server again it will send this token to the authorization header of the request, so the only thing needed for the server is to validate the signature. Resulting in the token being stored by the user and not by the server.

These authentication methods are a key factor when using sites like amazon or eBay, where it is key for the server to distinguish two different people and manage payment processes.

2.4.4.6 Other Options instead of an API

Another option would be using a direct connection between two points. It would certainly improve latency, but we would lose control of what is happening, customizability and it is risky in terms of security. Another issue would be the fact of losing the ability to know where an error is occurring, as in that case, an API can be redundant and solve issues on its own, instead of the need of a human.

2.4.5 Storage and databases

In a computer, there are two ways of storing information, the first and most common is in a disk drive, and the other way is using our RAM (Random Access Memory) which is in direct connection with our processor and enables us to save data which we can access almost instantly. The decision about which of them is better for each case is based on the needs:

- Disk Drives are slow but can store gigabytes of information without failure, these are most used in cases where the data is accessed less frequently and with security considerations. For a server it is common to use SQL (Structed Query Language) which organizes the data more efficiently.
- RAM is much faster than a disk drive, but it is also expensive and with little capacity in comparison with a disk drive. Furthermore, the data is stored in memory cells which on shutdown will wipe out, losing all the data.

2.4.5.1 What is a database?

A database is the part of a server where data is stored. This part is to be only accessible by the API, making it the most secure part of a server. Almost every database in the world is based on SQL or Structured Query Language, which is the current international standard for databases and has been around for some decades now as it is intuitive, simple, and efficient.

2.4.6 The Arduino

Arduino is one of the world's simplest and most reliable ways of processing and managing data at a simple level. It is used by thousands of people every day and helps create projects which need electronics and simple logic.

2.4.6.1 Usage of an Arduino

An Arduino is usually used to do simple logic and mathematical operations, but it can also be used as an intermediary between some Arduino sensors and a computer.
An Arduino is based on:

- Digital Pins, which are used to input and output voltage ranging from 255 (5V) to 0 (0V) which is considered a 0 or a 1 consecutively.
- Analog Pins have the same proprieties as a Digital pin, but with 10-bit resolution, so being able to return integers from 0 to 1023.

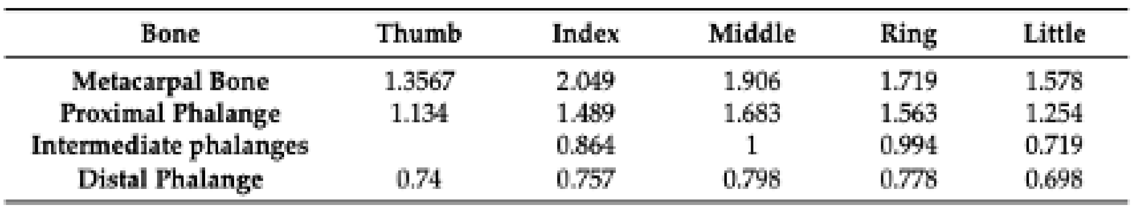
These two are used to compute and use the input information and transmit an output.

3. Research methodology: Hardware

3.1 Biomechatronic design

To start off with, a human hand anatomy-based design was used to create the prototypes. The starting point was from the inmoov hand, an open-source model created by Gael Langevin, which offered all the basic characteristics needed to get the project going. All the aspects and components of the prosthetic hand were developed and modified to mimic the dimensions, proportions, and functions of a human hand.

Before proceeding to the prosthetic hand design and assembly, the main design specifications based on the human hand behaviour as joins and movement capabilities were analysed to match the open-source hand. One of the main novelties of this prototype relies on the employed materials, which are ideal to reproduce human tendons, ligaments, fibrous sheaths, joins, etc. The prosthesis was build using polylactic acid or better known as PLA, a 3D printing material that is both strong, somewhat flexible, and easy to modify after printing. The bionic hand was designed using standardized bone values of an adult female (see Table 21). Bearing in mind that everybody is different, modifying these levels to match the user is done easily before printing. However, as 3D printing is required, these values slightly vary from the actual hand due to the need of supports, and all in all better printing environment.



Bone	Thumb	Index	Middle	Ring	Little
Metacarpal Bone	1.3567	2.049	1.906	1.719	1.578
Proximal Phalange	1.134	1.489	1.683	1.563	1.254
Intermediate phalanges		0.864	1	0.994	0.719
Distal Phalange	0.74	0.757	0.798	0.778	0.698

Figure 21 - Anatomic proportions of the human hand bones

As said before the human hand consist of various bones, tendons and cartilages etc. (see figure X) that make possible movement, without living out the interaction between muscles and nerves. We can see that every essential bone to allow simple movements is replicated, and therefore grasping and object and shaking hands is possible theoretically[.

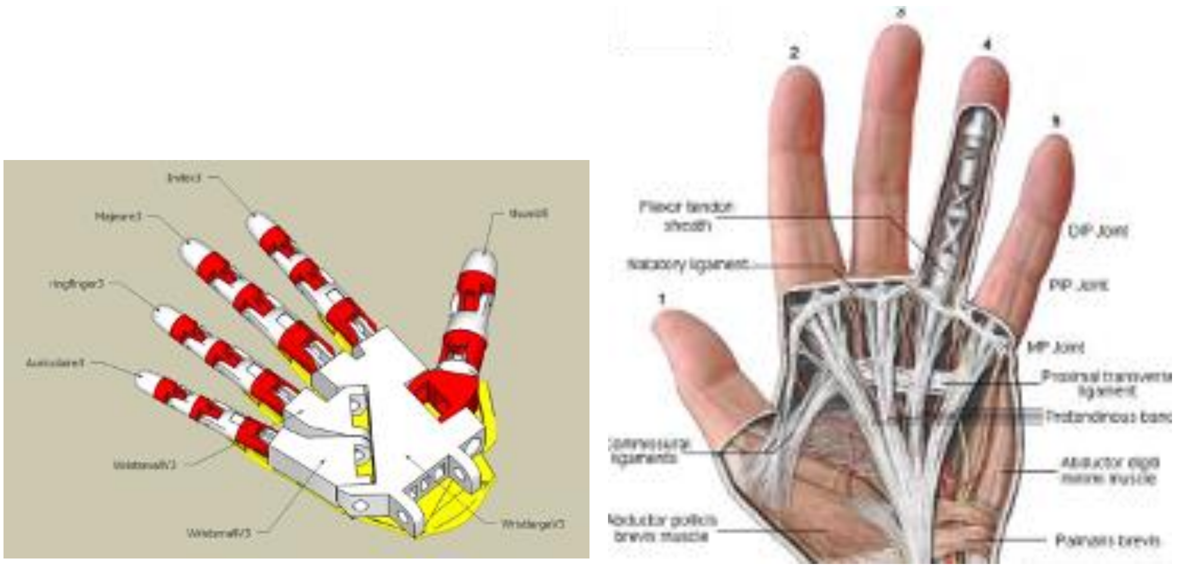


Figure 22 - showing the comparison of the 3D model and a human hand.

The proposed prosthesis was based on the idea of replicating human finger motions. The abduction/adduction, as well as its flexion/extension, are independent of hand anatomy. The solution for the joints is to design a volar plate and collateral ligaments (see Figure X a), controlled by the control system (see Figure 23)[14] [15] [16].

The kinematics of the index finger and extensor ligaments are completed using different materials (will be mentioned in the hand fabrication). Figure 24 shows the kinematics of the index finger and extensor ligaments. Five actuators/servos drive the five fingers, with one controlling each finger, each with its tendon line and route. Nonetheless, the phalanges' abduction/adduction movement is so slight that it makes little sense to measure

[14] Biryukova, E. V., & Yourovskaya, V. Z. (1994). A Model of Human Hand Dynamics. Advances in the Biomechanics of the Hand and Wrist, 107–122.

[15] Fok, K. S., & Chou, S. M. (2010). Development of a finger biomechanical model and its considerations. Journal of Biomechanics, 43(4), 701–713.

[16] *Valero-Cuevas, F. J. (2005). An integrative approach to the biomechanical function and neuromuscular control of the fingers. Journal of Biomechanics, 38(4), 673–684.*

it, the reason behind it is to avoid breaking and tearing of the line and 3D printed phalanges[17] [18] [19] [20].

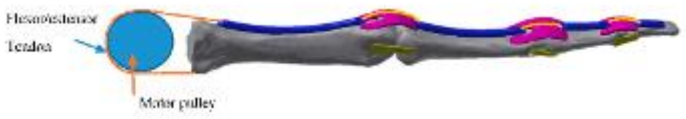


Figure 23 – Figure of a human finger robotized.

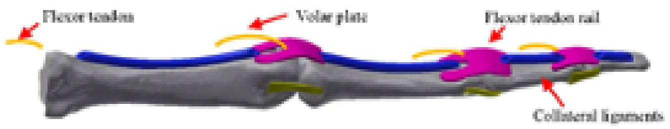


Figure 24 – Figure showing how a human finger act.

Following the hand comes the arm, in which all the components are stored to enable a fluid response. The 3D model (see figures from 25-30) were designed to allow both fluency and efficiency while moving. The weight of the hand and forearm is 700g, we are trying to reduce this weight to avoid problems when attached to the elbow mechanism, the one responsible for the actual shaking of hands[21] [22] [23].

[17] *L., J., Perez-Gonzalez, A., C., M., E., B., Vergara, M., L., J., … Morales, A. (2011). Towards a Realistic and Self-Contained Biomechanical Model of the Hand. Theoretical Biomechanics.*

[18] Cerveri, P., De Momi, E., Lopomo, N., Baud-Bovy, G., Barros, R. M. L., & Ferrigno, G. (2007). Finger Kinematic Modeling and Real-Time Hand Motion Estimation. Annals of Biomedical Engineering, 35(11), 1989–2002.

[19] Valero-Cuevas, F. J. (2009). A Mathematical Approach to the Mechanical Capabilities of Limbs and Fingers. Progress in Motor Control, 619–633.

[20] Parvini, F., & Shahabi, C. (2007). An algorithmic approach for static and dynamic gesture recognition utilising mechanical and biomechanical characteristics. International Journal of Bioinformatics Research and Applications, 3(1), 4.

[21]*Light, C. ., & Chappell, P. . (2000). Development of a lightweight and adaptable multiple-axis hand prosthesis. Medical Engineering & Physics, 22(10), 679–684.*

[22]*Cuellar, J. S., Plettenburg, D., Zadpoor, A. A., Breedveld, P., & Smit, G. (2020). Design of a 3D-printed hand prosthesis featuring articulated bio-inspired fingers. Proceedings of the Institution of Mechanical Engineers, Part H: Journal of Engineering in Medicine, 095441192098088.*

[23] *An, K. N., Chao, E. Y., Cooney, W. P., & Linscheid, R. L. (1979). Normative model of human hand for biomechanical analysis. Journal of Biomechanics, 12(10), 775–788.*

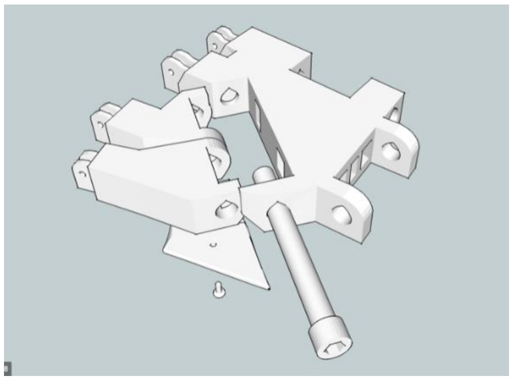


Figure 25 -Figure of one of the mechanisms of the hand

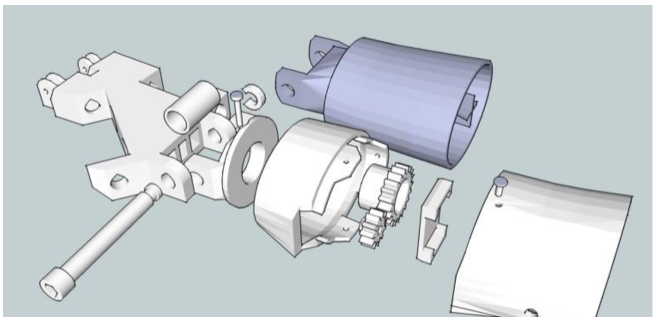


Figure 26 - Figure of the wrist mechanism.

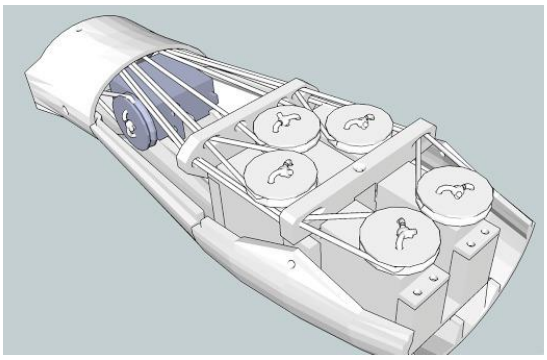


Figure 27 – Figure of the forearm with the wrist mechanism attached and the strings.

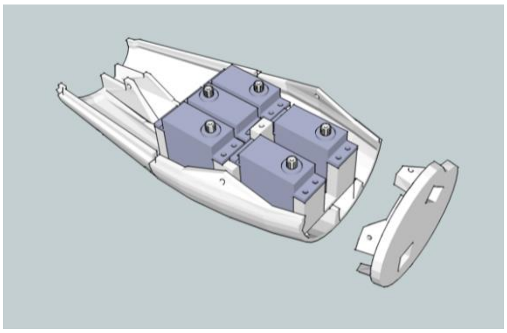


Figure 28 - Figure of the forearm with servos.

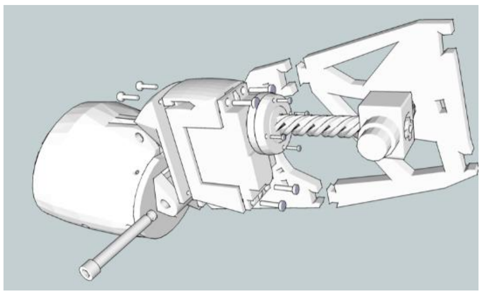


Figure 29 - Figure elbow mechanism.

3.2 Prototyping

We have done tree prototypes in total each one more complex than the previous one and with the improvements decided during testing.

The objective of the first prototype was to make a finger, just to see what would it take to be able to open and close. This prototype consisted of a small servo and an Arduino plus the 3d printed parts and the string.

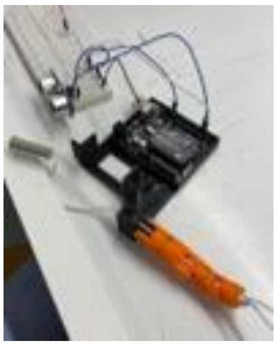


Figure 30 – Image of the first prototype

In the second prototype we added difficulty doing 5 fingers at the same time instead of one and substituting the small servo for five bigger and more powerful servos. The print was bigger, and the movement of the servos consisted in turning for closing the fingers and returning to the initial position for open them, so the strings were free to move and rubber bands that were in the back of the hand pulled the fingers to the initial position.

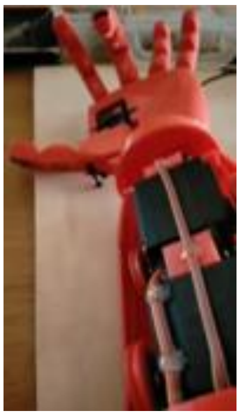


Figure 31 – Image of the second prototype

Finally, in the third prototype we 3d printed a bigger arm, replaced the rubber band mechanism with just one string that made all the route from the servo to the end of the finger and vice versa, so when the servo moved one string pulled and the other one pushed. In this prototype we also changed the battery for a bigger one and added a DC converter to regulate the power of the battery.

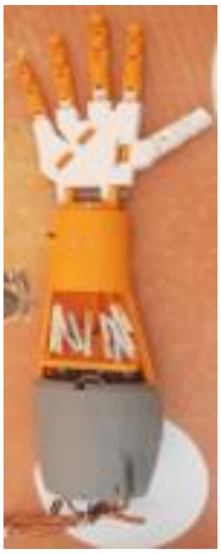


Figure 32- Image of the third prototype

4. Research methodology: Software

4.1 Introduction

For our project we need three steps: Sending, Storing and processing and finally, Receiving.

Each part will be custom made for our project and coded from scratch for our needs.

An overview:

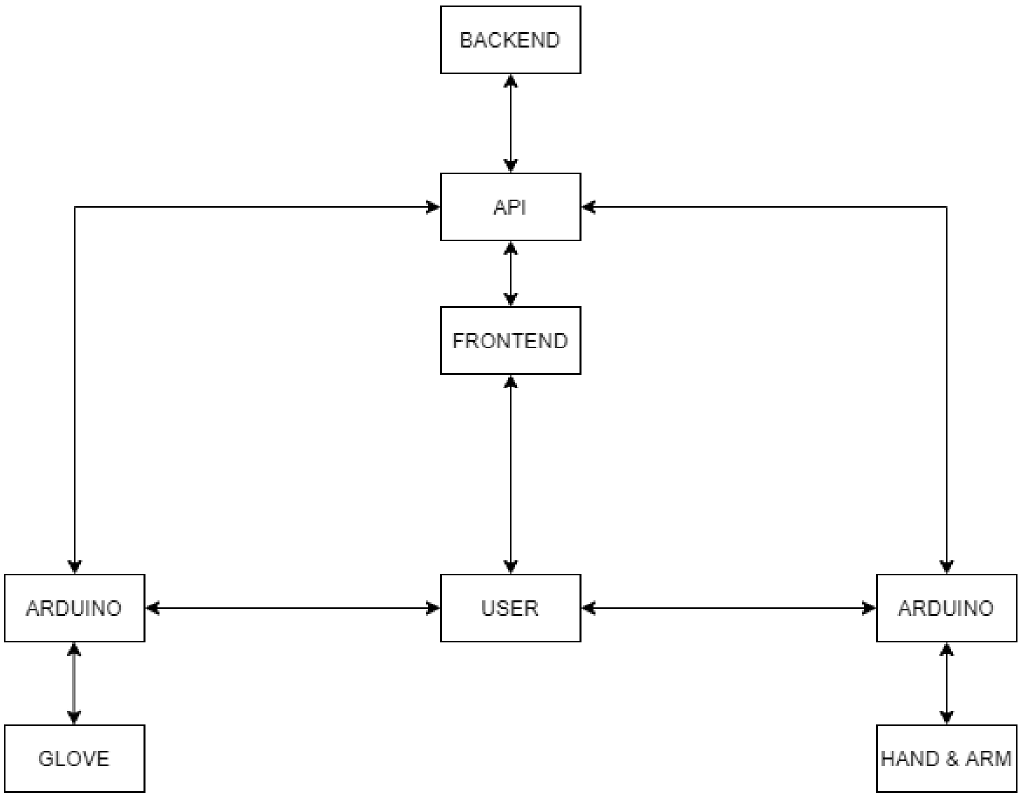


Figure 33 - Overview of the connection we have set up for this project

Source: Custom made

4.2 Our Frontend

4.2.1 Introduction

During our months of work, we have created two prototypes for our website, one made in Flask, and the other in Angular. The Frontend has expanded our possibilities, as it gave us the option of interacting with this process in a dynamic and prettier way.

4.2.2 Initial Approach

Our first webserver was made in flask, it was a simple one which was not finished, and it provided an initial approach to a Frontend and the coding involved. With that Framework we started designing the home page and the appearance of our website in general. It also was our first encounter with Web development and the start of a long learning curve.

4.2.3 Second and Main Frontend

Our second Frontend was created in Angular and with it we created our first public website with functionalities which enhanced our connections.

We distributed our Frontend into four main routes: Home Page, Hand Control, About and Contact Information. Each page contains the same navigation bar and social media, as Angular is a Single-page application, therefore we can conserve certain elements in each page and only change our main content.

Home Page Route:

Our home page route only contains the title and a button to redirect users to the About route where we explain our project, so the HTML code is fairly simple and there is no code running in the background.

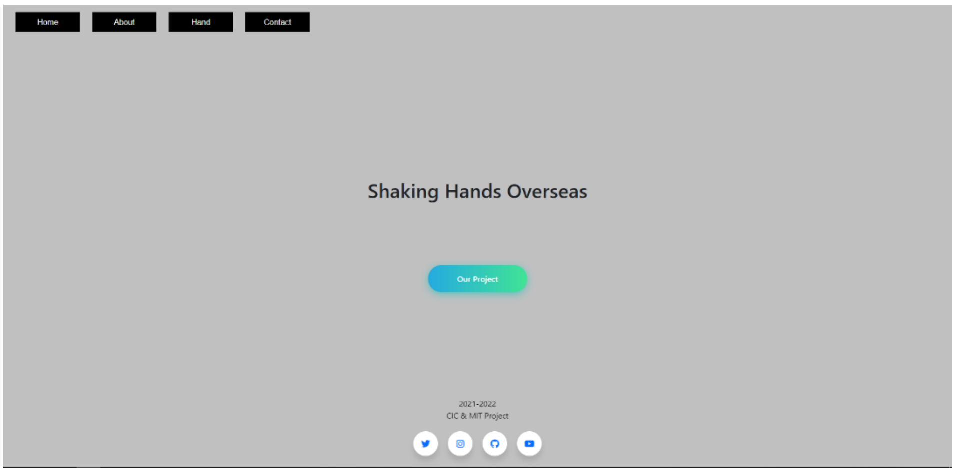
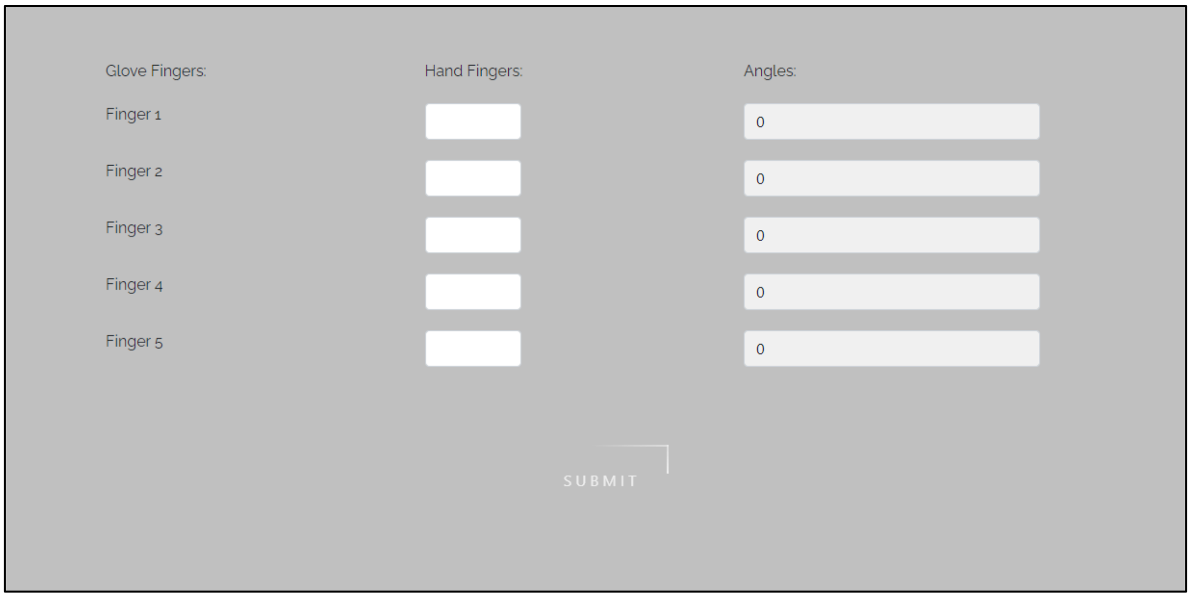


Figure 34 - Image of our Home Page in our Website

Source: https://shaking-hands-overseas-3f807.web.app/

Hand Control Route:



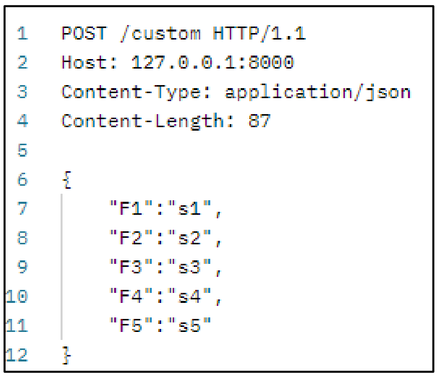
Glove Fingers:	Hand Fingers:	Angles:
Finger 1		0
Finger 2		0
Finger 3		0
Finger 4		0
Finger 5		0

Figure 35 - Image of our Hand Control Page in our Website

Source: https://shaking-hands-overseas-3f807.web.app/

In this image we can see the configuration page we have set up for our Hand Control Route, where we have a finger selector so that we can choose which finger will move and a custom angle selector, for testing cases, when we want to setup some servos in a concrete position. When you click submit, it will send two POST requests with a JSON file to our API as shown in the following image, where the "Host" is our API address, and in one it sends for each "F" (Short for finger) what sensor should be listening to, and the other, the numbers you have selected in the right-hand side of the page, which are sent as a sensor data to the API.

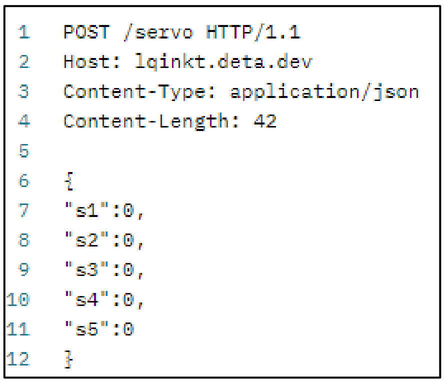
1.



```
POST /custom HTTP/1.1
Host: 127.0.0.1:8000
Content-Type: application/json
Content-Length: 87

{
    "F1":"s1",
    "F2":"s2",
    "F3":"s3",
    "F4":"s4",
    "F5":"s5"
}
```

Figure 36 - Image of a POST made to the "/custom" route on our server

Source: Postman

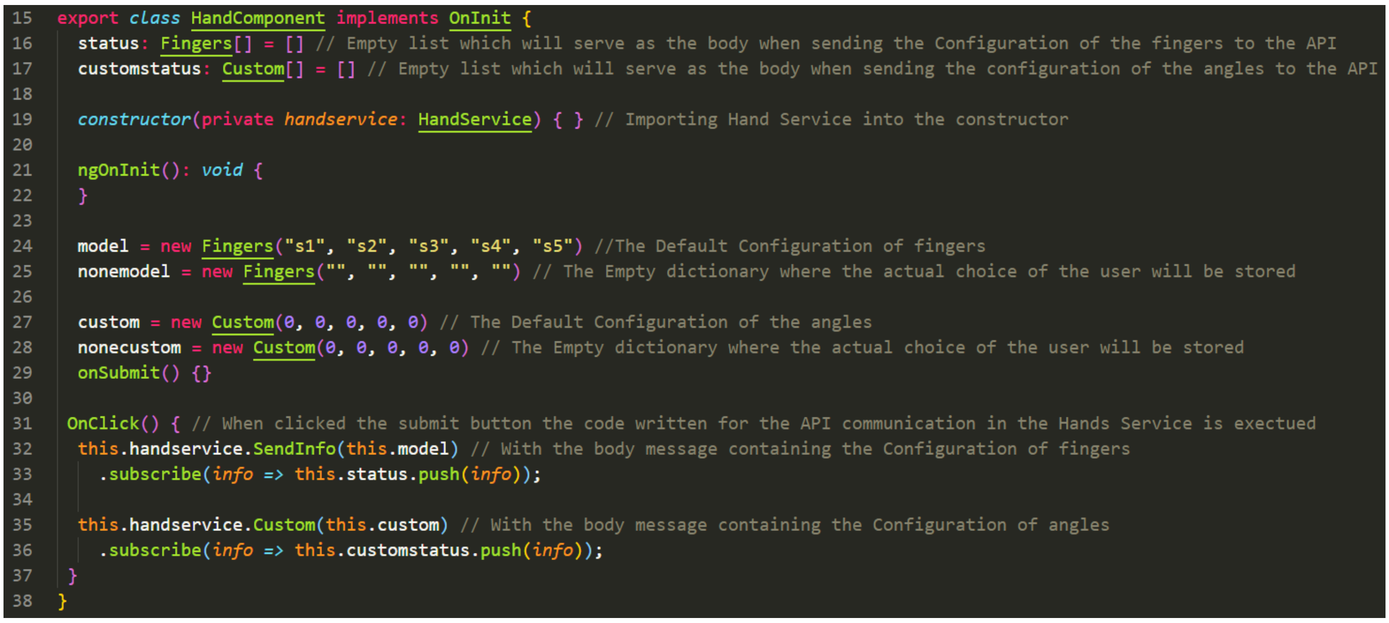
2. 

```
POST /servo HTTP/1.1
Host: lqinkt.deta.dev
Content-Type: application/json
Content-Length: 42

{
"s1":0,
"s2":0,
"s3":0,
"s4":0,
"s5":0
}
```

Figure 37 - Image of a POST made to the "/custom" route on our server

Source: Postman

This last request will be sent through the same route as the sensor arm sends its status, so it enables us to detect errors easily, as we can discard the API as the issue.

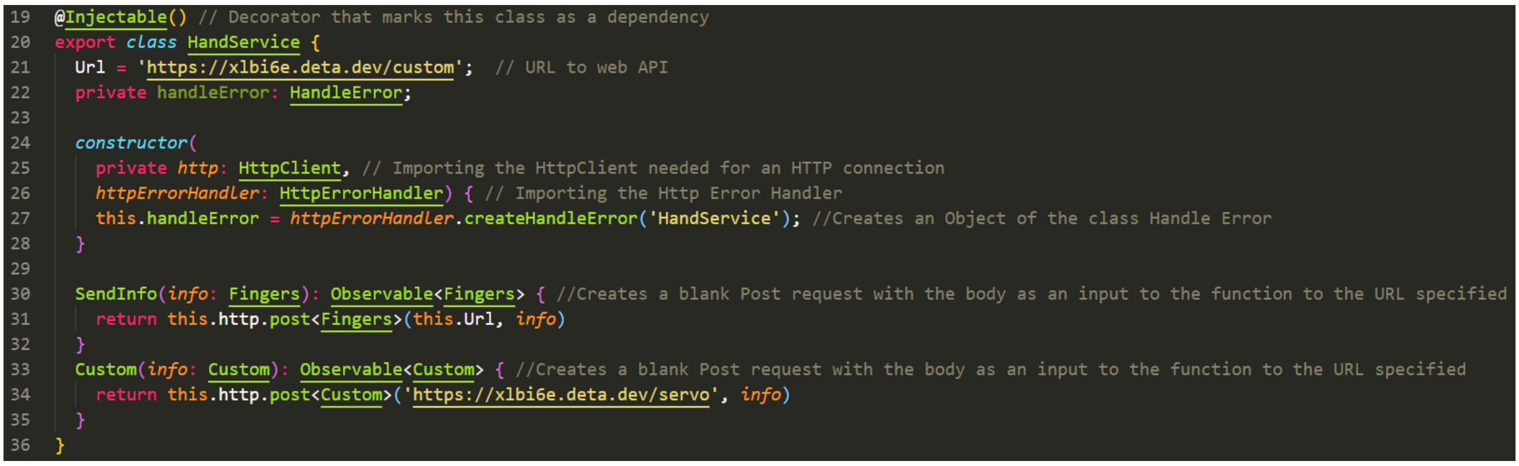


```
export class HandComponent implements OnInit {
  status: Fingers[] = [] // Empty list which will serve as the body when sending the Configuration of the fingers to the API
  customstatus: Custom[] = [] // Empty list which will serve as the body when sending the configuration of the angles to the API

  constructor(private handservice: HandService) { } // Importing Hand Service into the constructor

  ngOnInit(): void {
  }

  model = new Fingers("s1", "s2", "s3", "s4", "s5") //The Default Configuration of fingers
  nonemodel = new Fingers("", "", "", "", "") // The Empty dictionary where the actual choice of the user will be stored

  custom = new Custom(0, 0, 0, 0, 0) // The Default Configuration of the angles
  nonecustom = new Custom(0, 0, 0, 0, 0) // The Empty dictionary where the actual choice of the user will be stored
  onSubmit() {}

 OnClick() { // When clicked the submit button the code written for the API communication in the Hands Service is exectued
  this.handservice.SendInfo(this.model) // With the body message containing the Configuration of fingers
    .subscribe(info => this.status.push(info));

  this.handservice.Custom(this.custom) // With the body message containing the Configuration of angles
    .subscribe(info => this.customstatus.push(info));
 }
}
```

Figure 38 - Image of the code running in the background of the Hand Control Page in our website

Source: Our Code, Closed Source

This code is supported by the “Hand Service” dependency, which is another file where the code needed for this connection is written. The “Hand Service” code is described in the following image:



```
@Injectable() // Decorator that marks this class as a dependency
export class HandService {
  Url = 'https://xlbi6e.deta.dev/custom';  // URL to web API
  private handleError: HandleError;

  constructor(
    private http: HttpClient, // Importing the HttpClient needed for an HTTP connection
    httpErrorHandler: HttpErrorHandler) { // Importing the Http Error Handler
    this.handleError = httpErrorHandler.createHandleError('HandService'); //Creates an Object of the class Handle Error
  }

  SendInfo(info: Fingers): Observable<Fingers> { //Creates a blank Post request with the body as an input to the function to the URL specified
    return this.http.post<Fingers>(this.Url, info)
  }
  Custom(info: Custom): Observable<Custom> { //Creates a blank Post request with the body as an input to the function to the URL specified
    return this.http.post<Custom>('https://xlbi6e.deta.dev/servo', info)
  }
}
```

Figure 39 - Image of the dependency of the code running in the background of the Hand Control Page on our website

Source: Our Code, Closed Source

4.2.4 Hosting

For hosting our Frontend, we used Firebase from Google. The reason we chose this service is because of the integration with angular that it has, which facilitates our job when building the app and sending it to the cloud, because this is automatically done when running the angular build command.

4.3 Our Backend & API

For our overseas communications we need somewhere where we can store temporarily the status of the glove sensors, so that our hand can access them when needed. Here is where an API can help, being an intermediary between our hand and glove, where the API stores in its memory the last known position of each sensor and provides it to the other Arduino as soon as the request comes in.

4.3.1 Initial Approach

The first API we made was coded in python using flask and was used to support our Flask Website, but we discontinued this initial project when we started looking for a simple but effective way of doing a connection between Arduinos, as our Flask project was focused on supporting a Frontend and not an Overseas communication.

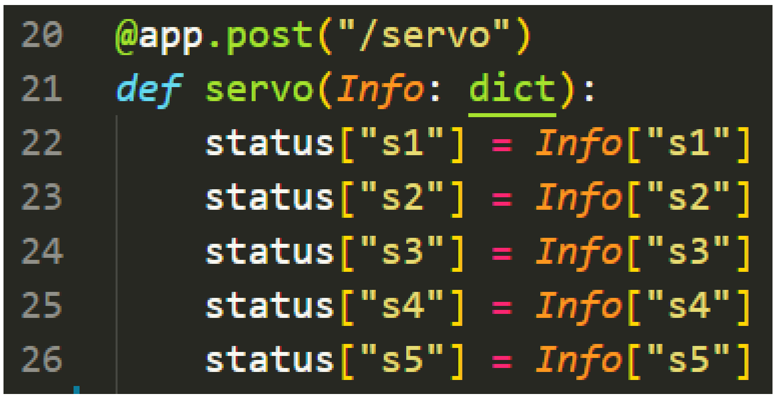
4.3.2 First Overseas Connection

When we wanted to do our first test overseas during the Edgerton Centre Workshop, we created an API using FastAPI, a basic API with limited but useful features. This was our first approach to a connection overseas and when we learnt all APIs Basic Functionalities. This API proved to function as intended and we were able to accomplish our goal of moving a finger from the other side of the ocean.

This API consisted of three main parts:

<u>[POST] Servo:</u>

The servo route is where the information is received and saved into a local dictionary called "status." The information is received in a JSON file which Is converted into a dictionary which python can use.



```python
@app.post("/servo")
def servo(Info: dict):
    status["s1"] = Info["s1"]
    status["s2"] = Info["s2"]
    status["s3"] = Info["s3"]
    status["s4"] = Info["s4"]
    status["s5"] = Info["s5"]
```

Figure 40 - Image of the “Servo” route of the code of the First API we made.

Source: Our Code, stored in https://github.com/Shaking-Hands-Overseas/Basic_API

<u>[POST] Custom:</u>

The custom route determines the order in which the data will be received from the server. So, in the case of connecting the sensor number 1 to the finger number 3, instead of changing the order in hardware, you can change it in the server. The order is stored in the same way as the sensor data is but instead of a number, the name of the data variable inside the data dictionary is stored in the order we want.

This functionality was added after the first demo when we introduced our Frontend.

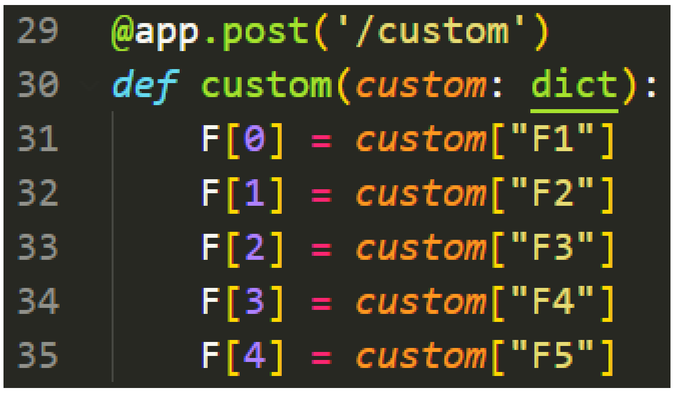

```python
@app.post('/custom')
def custom(custom: dict):
    F[0] = custom["F1"]
    F[1] = custom["F2"]
    F[2] = custom["F3"]
    F[3] = custom["F4"]
    F[4] = custom["F5"]
```

Figure 41 - Image of the “Custom” route of the code of the First API we made.

Source: Our Code, stored in https://github.com/Shaking-Hands-Overseas/Basic_API

<u>[GET] Reciever:</u>

The Reciever route is the responsible of sending back the information stored in the local dictionary. It sends back the information in the order stored In the “F” dictionary (Short for Finger).

Another great feature is the flexibility it provides, as we can change which sensor goes to each servo and store them if needed. So, then we have full control of our sensors and servos and an interface from where to work with.

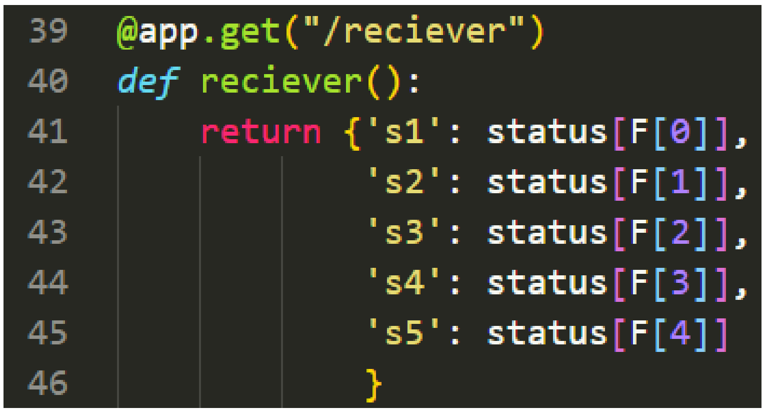

```python
@app.get("/reciever")
def reciever():
    return {'s1': status[F[0]],
            's2': status[F[1]],
            's3': status[F[2]],
            's4': status[F[3]],
            's5': status[F[4]]
            }
```

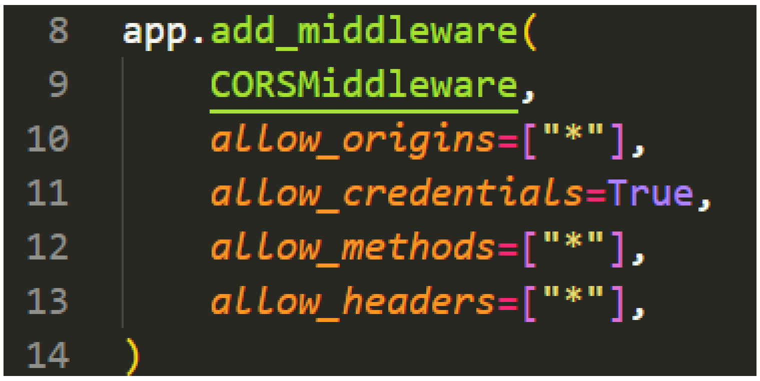
Figure 42 - Image of the "Receiver" route of the code of the First API we made.

Source: Our Code, stored in https://github.com/Shaking-Hands-Overseas/Basic_API

For this code to work properly, we needed to allow CORS middleware:

CORS being an HTTP header which allows the server to indicate any origins.

So, we are basically enabling communications which origin from anywhere and with whatever HTTP method and headers are used.



```python
app.add_middleware(
    CORSMiddleware,
    allow_origins=["*"],
    allow_credentials=True,
    allow_methods=["*"],
    allow_headers=["*"],
)
```

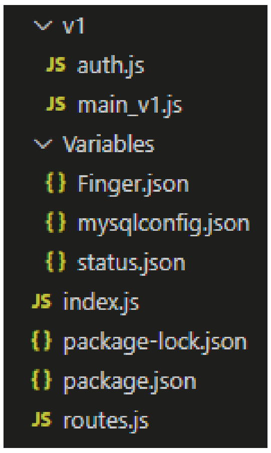
Figure 43 - Image of the CORS Middleware setting we have in our First API

Source: Our Code, stored in https://github.com/Shaking-Hands-Overseas/Basic_API

4.3.3 Our Second and current API

Our Second API and current API uses ExpressJS, which expands our possibilities, and it is more customizable. Our reason for changing it was the lack of control over the connections and their responses, as FastAPI was limited in that aspect. We also thought about having more than one connection at a time. For that matter, we needed to use a database or a different approach.

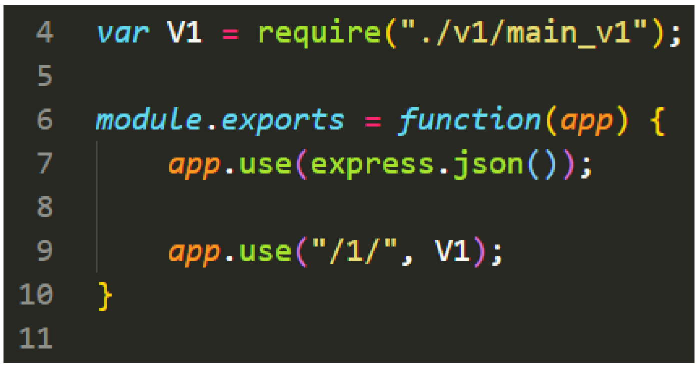
This API has the same three functionalities, as our objective is the same, but we started coding a module to connect with a database and a better folder structure, considering the possibility of having multiple versions of this API and having them working separately to avoid systems depending on an older version having trouble.



```
v1
  auth.js
  main_v1.js
  Variables
    Finger.json
    mysqlconfig.json
    status.json
index.js
package-lock.json
package.json
routes.js
```

Figure 44 - Image of the folder configuration we have in our Main AP.

Source: Our Code, stored in https://github.com/Shaking-Hands-Overseas/API

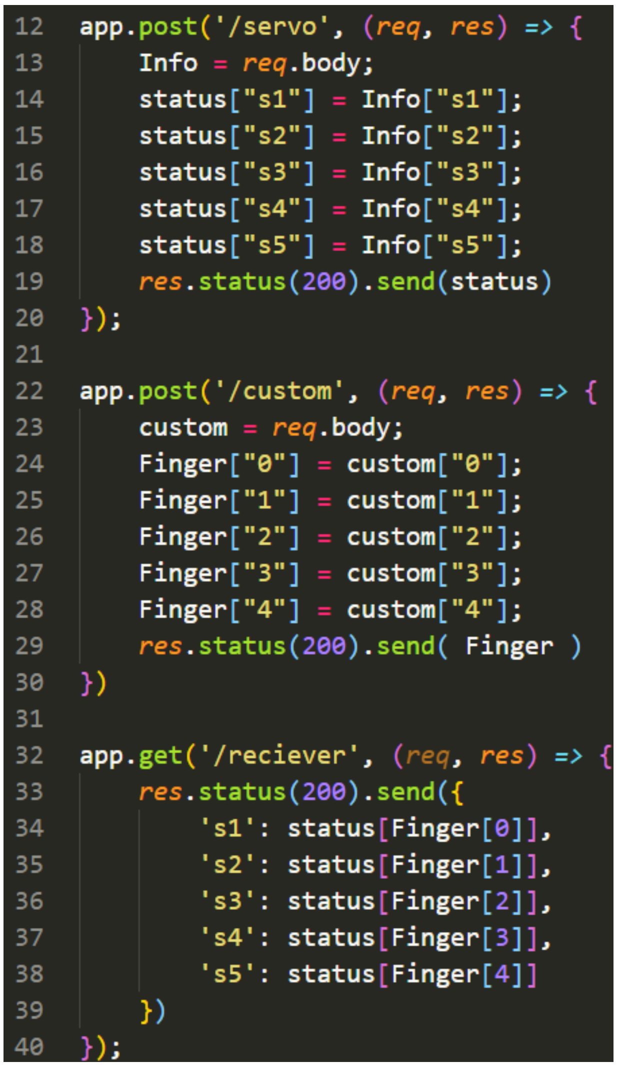


```javascript
var V1 = require("./v1/main_v1");

module.exports = function(app) {
    app.use(express.json());

    app.use("/1/", V1);
}

```

All these versions are imported into the main app by using the number indicating the version as part of their route. Consequently, if someone wants to access the second version, they must type …/2/… between the server DNS and the server route they are searching for.

Figure 45 - Image of the routing method used for all the API versions imported into the main app

Source: Our Code, stored in https://github.com/Shaking-Hands-Overseas/API



```
app.post('/servo', (req, res) => {
    Info = req.body;
    status["s1"] = Info["s1"];
    status["s2"] = Info["s2"];
    status["s3"] = Info["s3"];
    status["s4"] = Info["s4"];
    status["s5"] = Info["s5"];
    res.status(200).send(status)
});

app.post('/custom', (req, res) => {
    custom = req.body;
    Finger["0"] = custom["0"];
    Finger["1"] = custom["1"];
    Finger["2"] = custom["2"];
    Finger["3"] = custom["3"];
    Finger["4"] = custom["4"];
    res.status(200).send( Finger )
})

app.get('/reciever', (req, res) => {
    res.status(200).send({
        's1': status[Finger[0]],
        's2': status[Finger[1]],
        's3': status[Finger[2]],
        's4': status[Finger[3]],
        's5': status[Finger[4]]
    })
});
```

The programming of the routes in our new API does the same as the First API, but it is adapted to JavaScript, the programming language in which we are using now. As seen in figure 28, we also need to include the status code of the message in the response. In this case, we send a 200, as it is a successful connection. Also, ExpressJS lets us use the request information and tweak the response as we prefer, something our First API does not allow us to do in a simple manner.

Figure 46 - An overview of the three main routes of our second and current API

Source: Our Code, stored in https://github.com/Shaking-Hands-Overseas/API

4.3.4 Our Database

A database is useful to store big amounts of data in an organized manner, so the initial part of this project has not been used. Although we have planned its usage in the future to store data and create preprogrammed movements.

4.4 The Arduino

4.4.1 Introduction

The Arduino is the responsible of connecting the sensors/servos with a computer, and in this part, we will explain how this works to ensure all data is as we want it to be.

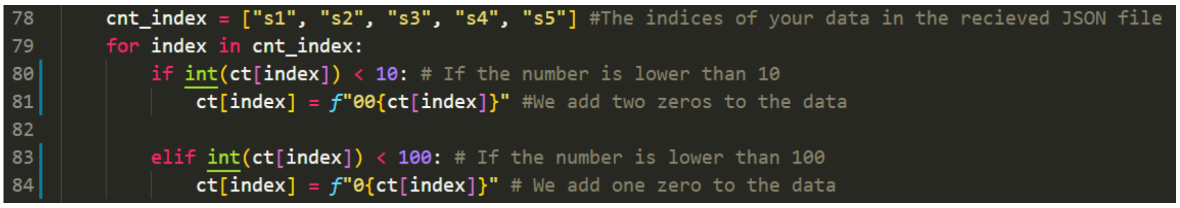
4.4.2 Initial Approach

When we first tackled on the matter, we knew we had to send data to a server, and we did not know how. Our first idea was using an Arduino Wi-Fi Module, with which we spent some days on, but this approach proved to be ineffective due to the lack of documentation and the module itself malfunctioning. Consequently, we decided to ask for help to our MIT colleagues, and a couple of days, we finally decided changing our approach to this communication, as it proved to be harder than expected.

4.4.3 First Working Connection

Our second approach was provided by asking Arnau Ortega, who was a student from our school and who was working as a mentor in the Egerton Centre Workshop. He gave us the idea of using the serial monitor for data transmission and reading it via a python module named PySerial. Using this module, we had to find the best way of sending and receiving the data, and after some days of testing sending all data separately, we found out that sending all the data within a single string was the best option. This required encoding the data so that it would not be misinterpreted. After some tweaking, the best option we found was encoding it as follows:

- The first step was checking how many digits each number had. So, checking if the number consisted of more than a single digit, as for our data transmission, we needed to send a constant String with invariable length. To accomplish this, we created a simple logic so that if the number received was smaller than 10, we added two zeros and if the number was smaller than one hundred, we only added one. If the number is above one hundred, we do not need to do anything, as the data we are working with ranges from 0 to 180. With this we know for certain that each sensor data will not be larger than 3 sdigits.

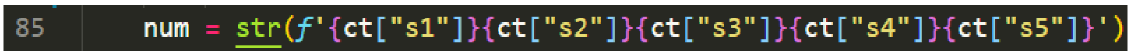


```
    cnt_index = ["s1", "s2", "s3", "s4", "s5"] #The indices of your data in the recieved JSON file
    for index in cnt_index:
        if int(ct[index]) < 10: # If the number is lower than 10
            ct[index] = f"00{ct[index]}" #We add two zeros to the data

        elif int(ct[index]) < 100: # If the number is lower than 100
            ct[index] = f"0{ct[index]}" # We add one zero to the data
```

Figure 47 - Logic that we have implemented to send a constant length string

Source: Our Code, stored in https://github.com/Shaking-Hands-Overseas/Arduino-Serial-Hand

- Then putting all the data compacted into a single string without spaces.

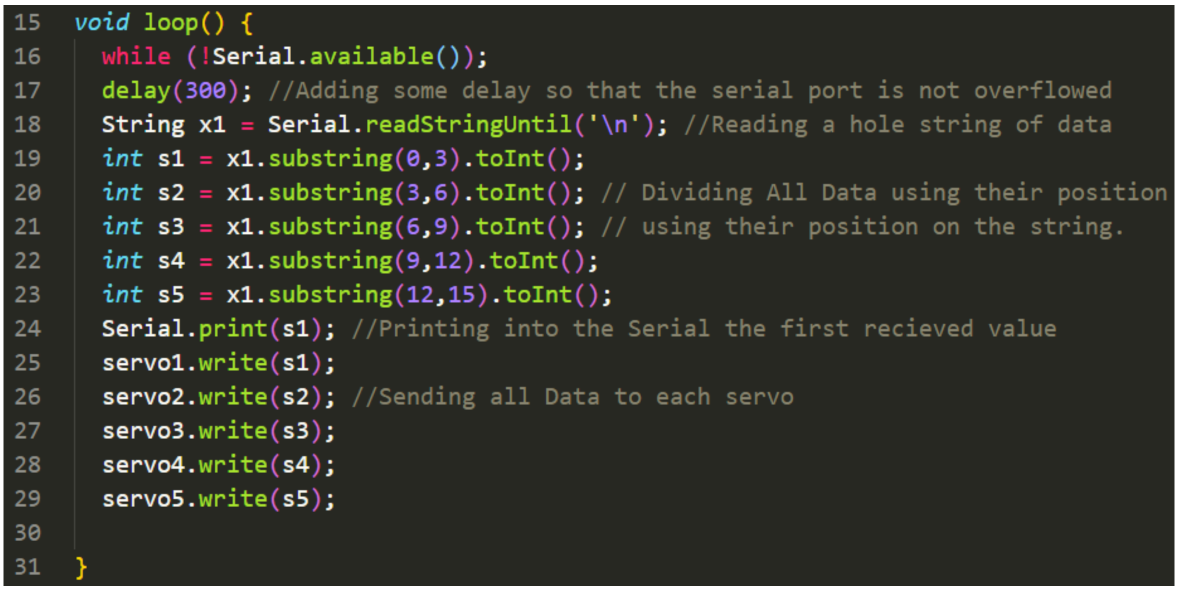


```
    num = str(f'{ct["s1"]}{ct["s2"]}{ct["s3"]}{ct["s4"]}{ct["s5"]}')
```

Figure 48 - String that will be sent into the Arduino with constant length

Source: Our Code, stored in https://github.com/Shaking-Hands-Overseas/Arduino-Serial-Hand

- The next step is sending the data to the Arduino. Once the Arduino received that string, it will be decomposed into parts by diving the string into each sensor reading. As the string length is constant, we only need to divide it into the data we want using the position on the string. Just before receiving the data, the Arduino will respond sending the data from the first sensor that it has received. This was created to check if the data was being sent as intended.



```
void loop() {
  while (!Serial.available());
  delay(300); //Adding some delay so that the serial port is not overflowed
  String x1 = Serial.readStringUntil('\n'); //Reading a hole string of data
  int s1 = x1.substring(0,3).toInt();
  int s2 = x1.substring(3,6).toInt(); // Dividing All Data using their position
  int s3 = x1.substring(6,9).toInt(); // using their position on the string.
  int s4 = x1.substring(9,12).toInt();
  int s5 = x1.substring(12,15).toInt();
  Serial.print(s1); //Printing into the Serial the first recieved value
  servo1.write(s1);
  servo2.write(s2); //Sending all Data to each servo
  servo3.write(s3);
  servo4.write(s4);
  servo5.write(s5);

}
```

Figure 49 - Loop function located on the Arduino firmware of our hand

Source: Our Code, stored in https://github.com/Shaking-Hands-Overseas/Arduino-Hand-Firmware

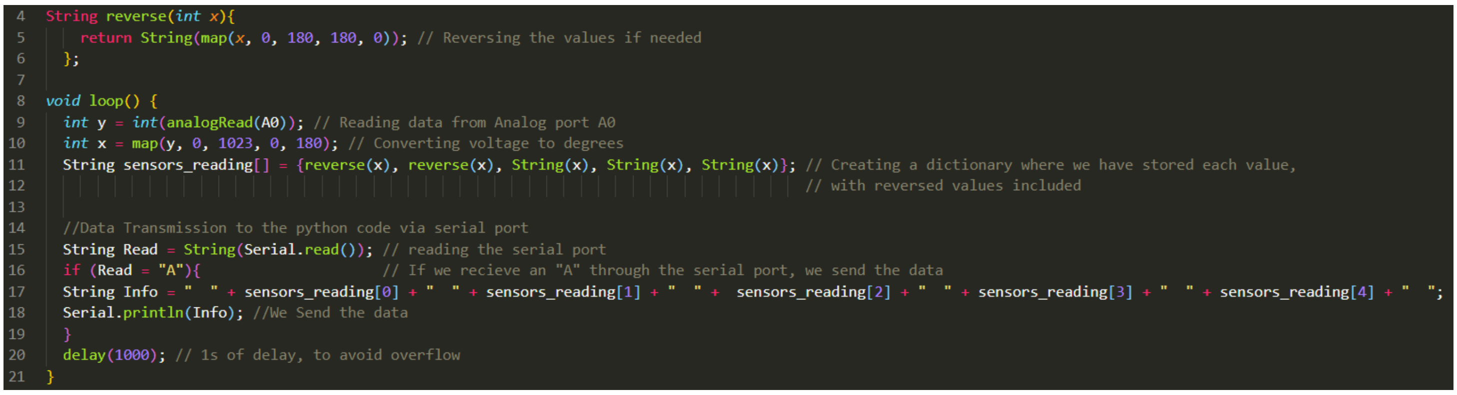
With this code, we were able to move a single finger in our Prototype hand, where the request to the API was sent by Edward Moriarty, who was located in Boston. This proved we were able to do this type of connection.

4.3.4 Connection using a potentiometer

As a second test, we used a potentiometer connected to another Arduino, which sends the position to a computer, and then the computer sends it to our API. We used this potentiometer to move the whole hand at the same time. The receiving and moving code of the hand is the same as in the First Overseas connection.

For this connection we read the status of the potentiometer via the analog port on the Arduino board and using the Arduino map function, we convert the values from the interval [0, 1023] to the interval [0, 180]. So, changing the information from Volts (0 representing 0V and 1023 representing 5V) to degrees for the servos. We also had to consider that some servos were inverted due to their position in our prototype. Consequently, we inverted them using another map function.

The last step is sending this data to our computer, to accomplish this, we used a simpler way than in our last connection. Instead of maintaining a constant length in a string, we include spaces in the message. So, when we receive the string, we divide the data using the split function in python.



```
String reverse(int x){
    return String(map(x, 0, 180, 180, 0)); // Reversing the values if needed
  };

void loop() {
  int y = int(analogRead(A0)); // Reading data from Analog port A0
  int x = map(y, 0, 1023, 0, 180); // Converting voltage to degrees
  String sensors_reading[] = {reverse(x), reverse(x), String(x), String(x), String(x)}; // Creating a dictionary where we have stored each value,
                                                                                        // with reversed values included

  //Data Transmission to the python code via serial port
  String Read = String(Serial.read()); // reading the serial port
  if (Read = "A"){                      // If we recieve an "A" through the serial port, we send the data
  String Info = "  " + sensors_reading[0] + "  " + sensors_reading[1] + "  " +  sensors_reading[2] + "  " + sensors_reading[3] + "  " + sensors_reading[4] + "  ";
  Serial.println(Info); //We Send the data
  }
  delay(1000); // 1s of delay, to avoid overflow
}
```

Figure 50 - Image where we observe the process of interpreting the data from a potentiometer and sending it to a computer

Source: Our Code, stored in https://github.com/Shaking-Hands-Overseas/Arduino-Glove-Firmware

In our computer, we will execute a similar code as in the First Overseas Connection, but instead of receiving data, it will send it with a POST request containing the data we received from the Arduino.

5. Shaking hands overseas

Our last test was the actual shaking hands, which was made on the 8th of April 2022 with LucasVR. LucasVR or Lucas, is an MIT freshman who has been working for the last year on a sensor glove to move stuff in virtual reality. We have talked with him for months, modifying his glove to our needs, to prepare what happened that 8th of April in which we accomplished moving all fingers of our hand with his sensor glove in real time.

Figure 51 Photo of Luca's glove when we were shaking hands

Figure 52 Photo of our hand when we were shaking hands

Source: Recording of the shaking hands on the 8th of April 2022

Link : (https://drive.google.com/drive/folders/1HiH8SVVHWO348zEFesq7a5lO12sEKhrA?usp=sharing)

As Luca's glove is being recreated all over the world by teenagers' workshops, this demonstration has enabled us to connect with other parts of the world and planning shaking hands with Ferrara in italy, UK,…

6. Recreating our project & Last Prototype

To build our last prototype, we will use these materials:

- 180 degrees servos
- Arduino power supply / 12V to 5V converter + 12V battery
- 3D printed hand
- Arduino Uno or ESP32
- Lubricant
- Fishing string

We will start by mounting the hand. In our last prototype, we only used the hand itself and the servo mount, as the rest of the hand and arm are redundant (We decided to remove all redundant parts and simplify the prototype). To mount it you should stick with hot glue both the hand and servo mount into a piece of wood or PLA.
Once you have mounted the hand, we will use Fishing String to move the fingers, so we have to put them in and with them we will connect the fingers with the servos.

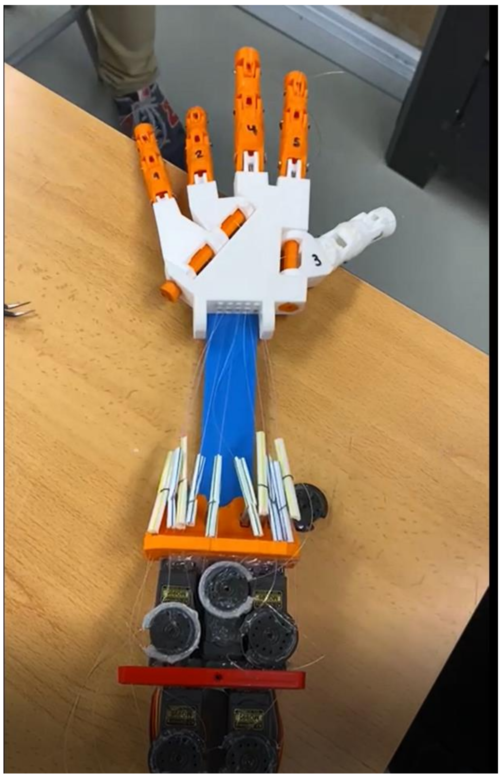


Figure 53 Photo of our final prototype

Once we have set up the hand, we will proceed to plug in the servos to the Arduino. This Schematic shows our cable management and how it should be done.

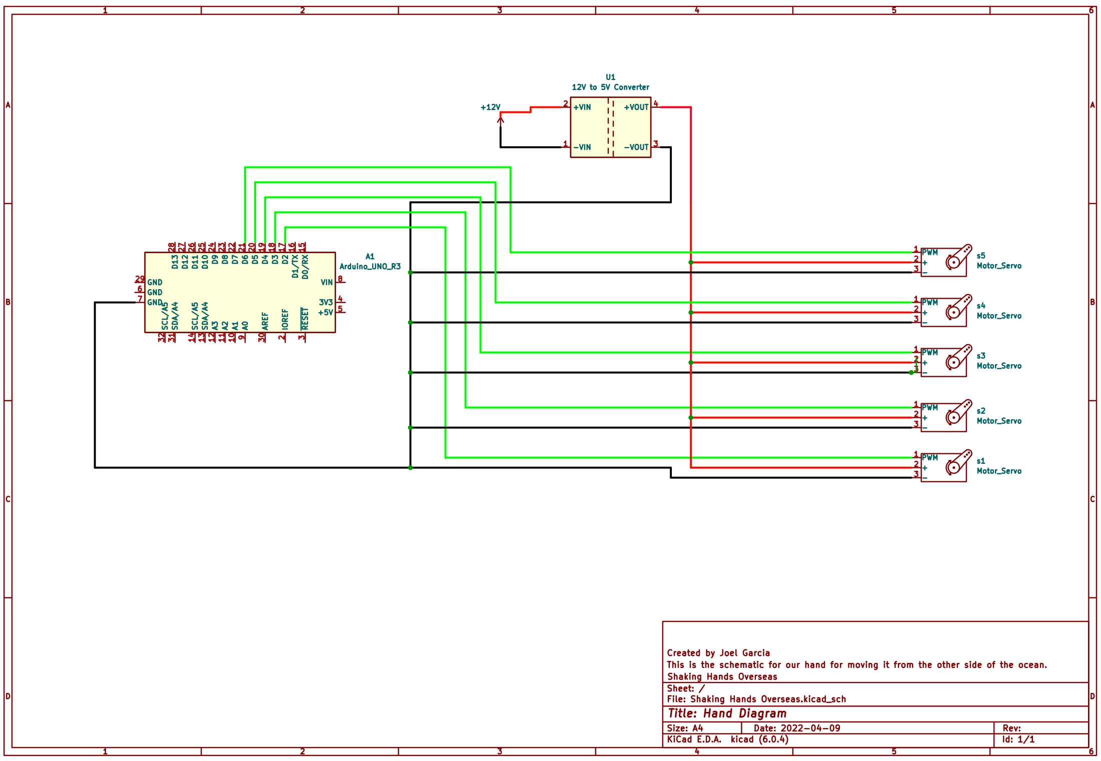
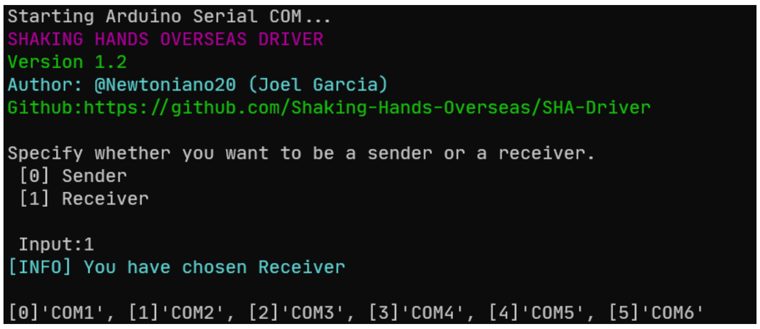


Figure 54 Schematics made by us showing how we cable managed our prototype

Once we have done all the hardware, we will proceed to the software, where we need a server running our API code. For our case we used Heroku[24], a free API hosting service, and we used the code stored in our GitHub[25]. Then, we will proceed to upload the firmware to our Arduino, which is also stored in our GitHub[26].

Now, we only have to download the driver[27] and select receiver mode and the serial port in which the Arduino is connected.



```
Starting Arduino Serial COM...
SHAKING HANDS OVERSEAS DRIVER
Version 1.2
Author: @Newtoniano20 (Joel Garcia)
Github:https://github.com/Shaking-Hands-Overseas/SHA-Driver

Specify whether you want to be a sender or a receiver.
 [0] Sender
 [1] Receiver

 Input:1
[INFO] You have chosen Receiver

[0]'COM1', [1]'COM2', [2]'COM3', [3]'COM4', [4]'COM5', [5]'COM6'
```

Figure 55 Screenshot of our driver mode selector

[24] https://dashboard.heroku.com/
[25] https://github.com/Shaking-Hands-Overseas/API
[26] https://github.com/Shaking-Hands-Overseas/Arduino-Hand-Firmware
[27] https://github.com/Shaking-Hands-Overseas/SHA-Driver

Once you have done that, your hand should start moving as you want. The construction of the glove has been covered by Lucas (Its creator) and it can be seen in his website[28], as it is not a part of our project although we have built one.

[28] https://hackaday.io/project/178243-lucidgloves-vr-haptic-gloves-on-a-budget

Results

The experiments carried out with the prosthetic hand aimed to verify the correct functionality of bones and tendons/wires. Moreover, in these experiments, the hand was also subjected to software tests (or unit tests). It is important to analyze the correct fingers flexion/extension trajectory because it is the one that can give us the most information on how does the hand work.

The main results are as follows:

- The basic grasping time was 1.3 s from the open hand position. This includes the sum of all latencies in the system, plus the servos' 0,51s for 180 degrees of movement.

- For a better object manipulation, independent phalanges and tendons that can be controlled by the same actuator incrementing joins kinematics and mechanics are necessary.

- The hand was evaluated and tested for 6 months, being subjected to changes to improve controllability end efficiency. During this period, multiple grasping movement experiments were performed, including the shaking hands overseas demonstration explained before.

- Some disadvantages were observed during testing to solve them, different strategies are set in place and will be tested in a short future.

The main problem we have now is the use of the servos. Their weight, size and power needs make the prosthetic hand dependant on the same problems as the state-of-the-art prosthesis do. Finger load is correlated with the rate torque of the actuators

During these tests, we made the following graphics in which we analyse the different latencies we must bear in mind during an overseas connection.

To start with, we checked the latency in the data transmissions between the Arduino and our computer during a hundred connections made milliseconds apart. After these connections we created the following graph:

In all the following graphs, the x values represent in which consecutive connection we are on, and the y values represents the latency in seconds.

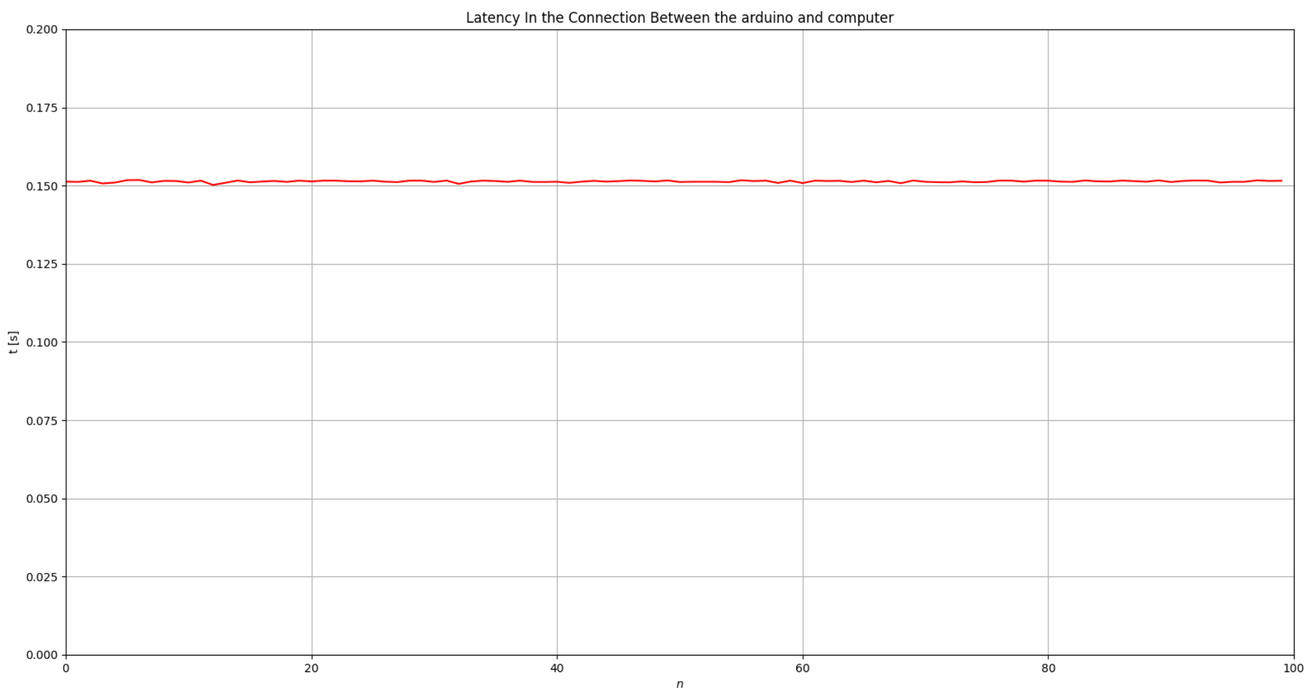


Figure 56 - Graph plotting the latency in the connection between our Arduino and computer

Source: Custom made

Later we did the same but with the latency between us sending a GET and POST request to our API and receiving it. We did 50 measurements, which were distanced milliseconds apart. After plotting all the data we came up with the following graphs:

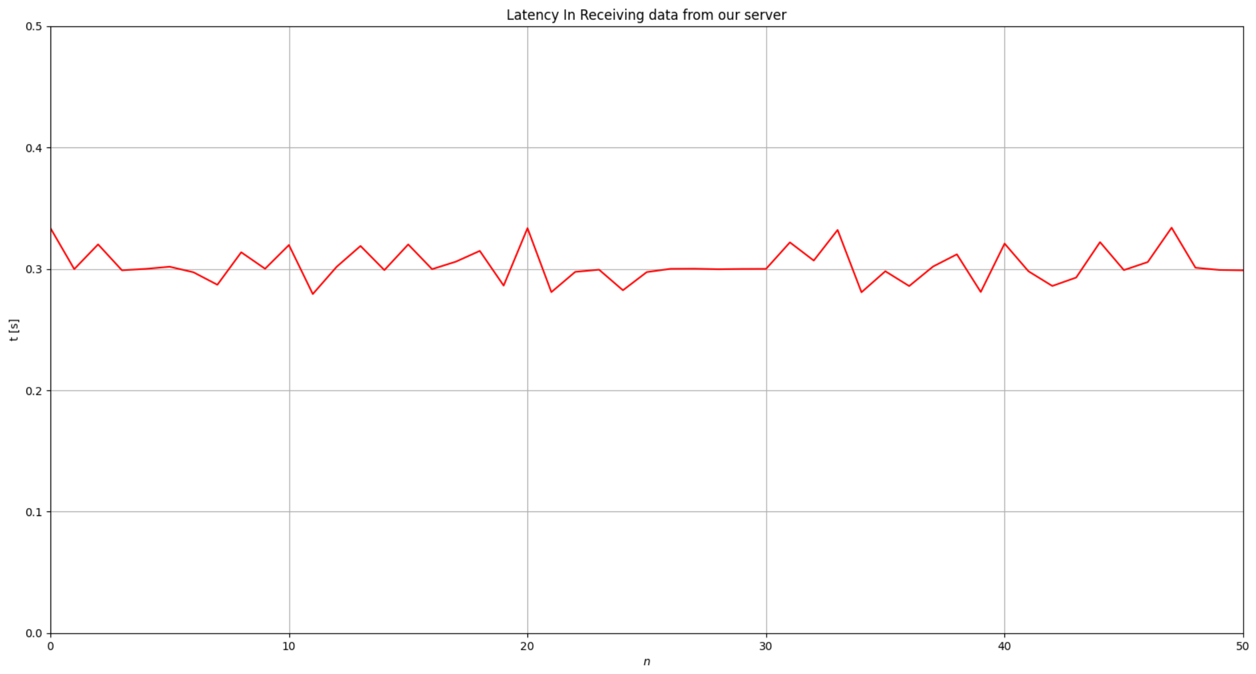


Figure 57 – Graph plotting the latency between us sending a GET request from our server and us receiving the requested information.

Source: Custom made

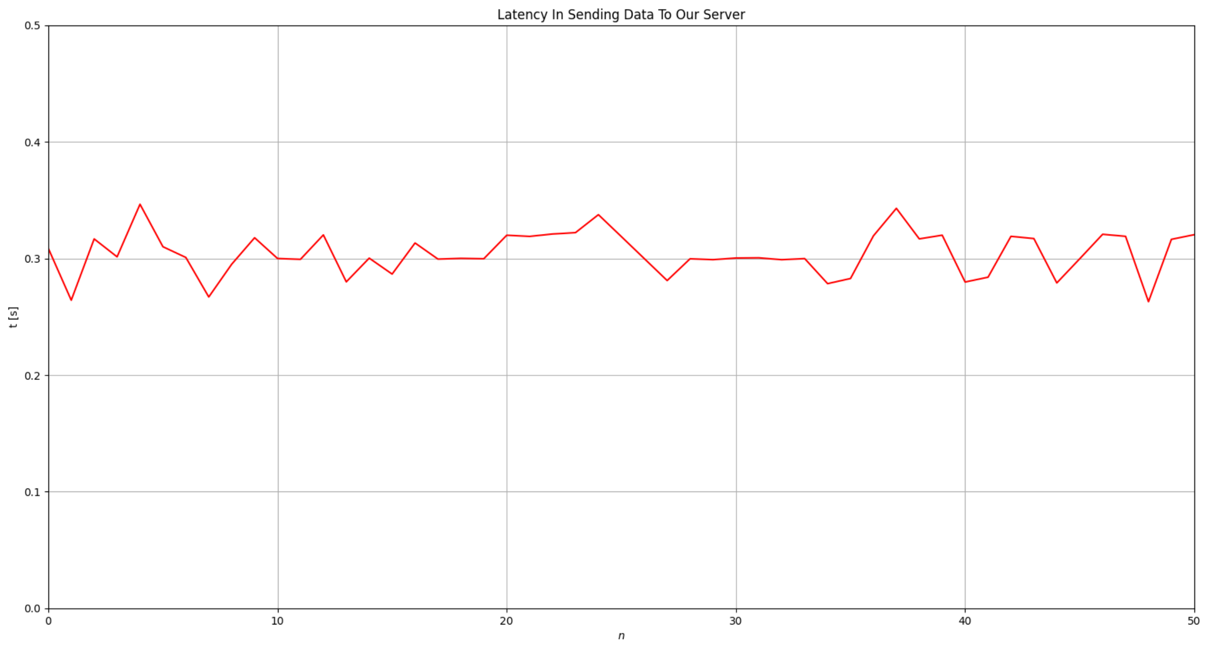


Figure -58 Graph plotting the latency between us sending a POST request to our server and us receiving the response from this server.

Source: Custom made

Using this data, we found that there was no penalty on latency after a particular number of connections, consequently, we had no data overflow. Also, we found that a connection with our server has a latency of 0.3 and that it does not depend on the method used in our API.

At the end, we measured the latency of our receiver code, which reads the data from our server and sends it to our Arduino. And we ended up with the following graph:

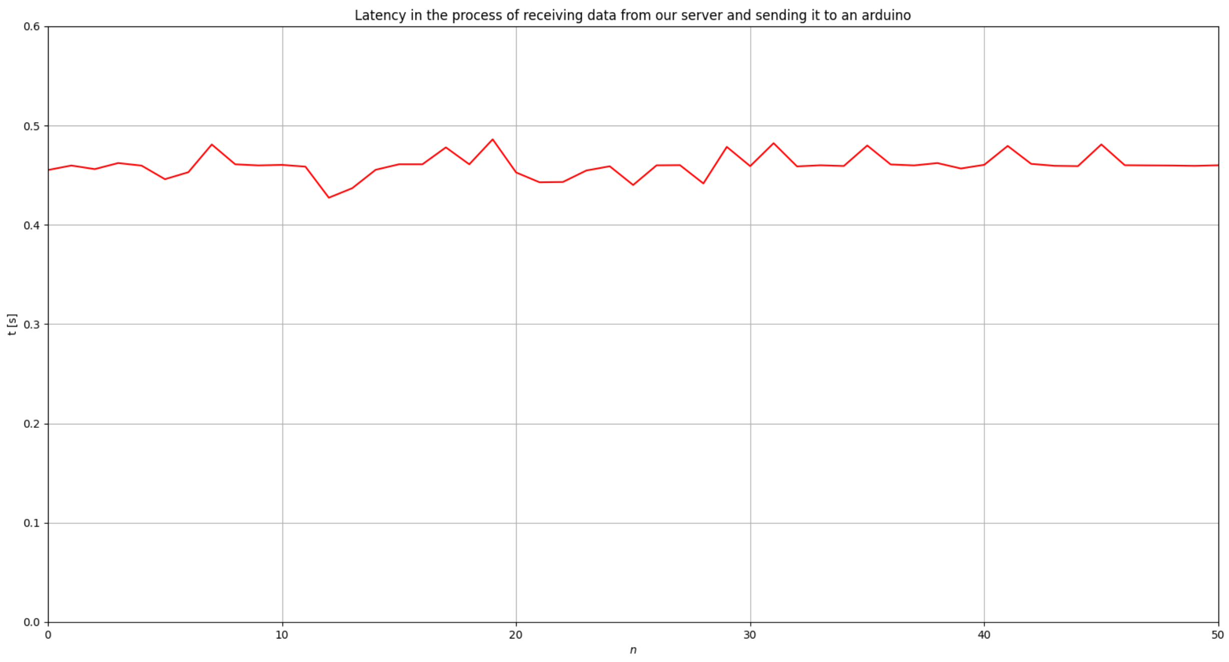


Figure –59 Graph plotting the latency between us requesting information to our server and sending it to our Arduino successfully

Source: Custom made

Using this graph we validated our past graphs, as this one is quite close to 0.45 seconds of latency, what happens to be the sum of the latency on a connection with our server (0.3 seconds) and the latency on our Arduino (0.15 seconds).

Finally, we measured the time it takes for a human to notice a movement in our hand and plotted into a graph. This was made to check if our last measurements where correct if when changing the angle of a finger with our webpage. If our calculations are correct, we should get a latency of approximately 0.8 seconds without counting human perception, as the sum of a POST request (0.3s) (We are also sending data, not only receiving), a GET request (0.3s) and the Arduino (0.15s) equals 0.75 seconds. The result we got is the following graph:

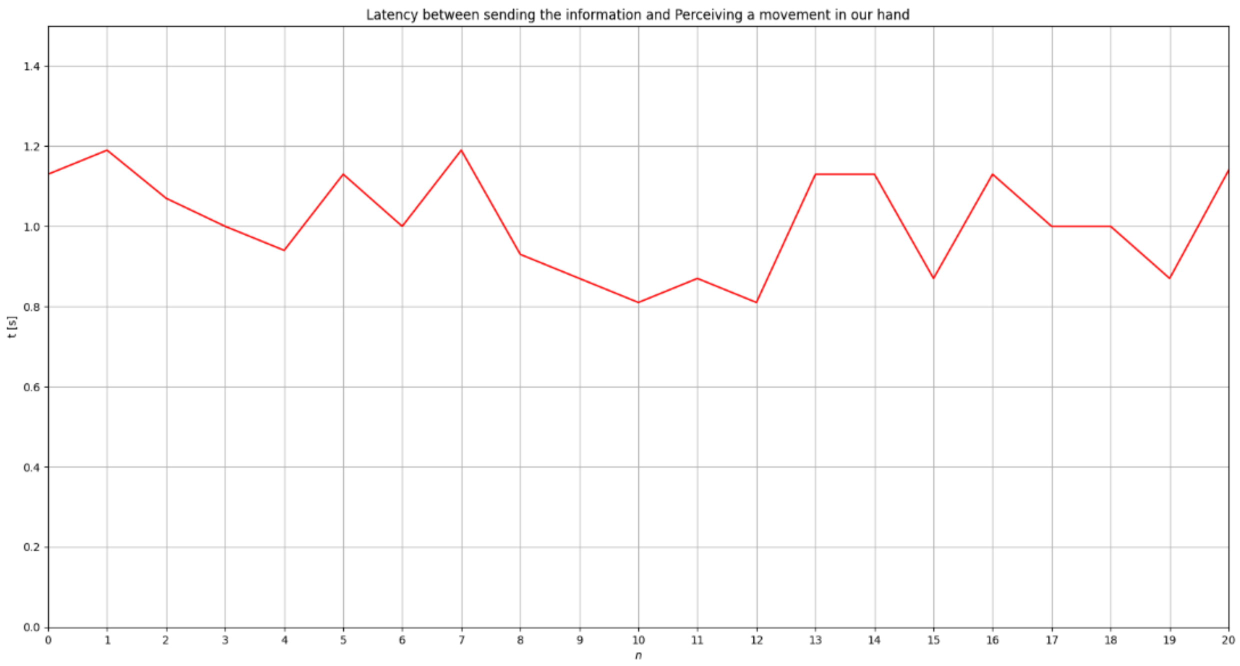


Figure –60 Graph plotting the human perception time between sending data to our server and perceiving a movement in our hand.

Source: Custom made

This graph validated all graphs, as we see 1 second of latency between sending information to our API and perceiving a movement in our hand. If we subtract our initial prediction and our result, we get 0.25 seconds (1 – 0.75), which represents the human perception latency on average following research in videogames and human perception[29].

[29] https://www.pubnub.com/blog/how-fast-is-realtime-human-perception-and-technology/

Conclusion

In this research paper, a different approach to human contact was presented. The use of a biomechanical arm interacting with a movement sense system, enabled us to shake hands overseas and connect with people around the world. After this system was subjected to tests some conclusions can be drawn:

A) Looking back at the project. We can affirm that we have been able to reach our goal of making possible an overseas human synthetic interaction and that these 8 months of work have been put to good use. Throughout the project we faced numerous difficulties we had to overcome which we learned a lot of. If we had to start again, we would have more time not to only focus on the building of the hand but on applications for it, as we would build it much faster. One this projects we would've liked to apply our hand would be teaching it to talk through hand signals. And the list goes on.

B) Being more specific. Our final model of the hand has been successful with the initial objectives but with a lot of limitations in movement and precision. We of course had a lot of other ideas and concepts we would have liked to try to implement to our final product. That is why we will most likely keep working on the project from time to time to see how far we can go.

C) As for the software part, we could argue it has achieved most desirable result. Even though there are countless improvements to be made, their repercussion is not that meaningful for all the work it requires. Although it is true, that we are left with a little bitter taste regarding the latency of response of our hand. Because as you could expect we would have liked it to be almost instant to achieve this a more natural interaction. Even though, this does not depend as much on us as it does on our equipment. Meaning that with more founding this could become a reality.

D) As for our personal level, we can assure we have grown as engineers and as people. We now really know what it involves to design and produce a working product and all the processes and steps that should be taken in order to make it a reality. As well as, that at end if you are looking to achieve something you can make it true if you really put the work it needs. Because from the first moment we would not have imagined what we have accomplished.

Future of the project

We have been working on this project for almost a year now and we are determined to continue this project for as long as we can. We are excited to keep improving our prototype and finding new ways of doing things.

An example would be that we are currently working on force feedback in the connection between the hand and the sensor glove. This would enable more dynamic connections and the ability of grabbing something with it and interacting with the environment. We are currently on the prototyping part of the latter and we are convinced it can be made.

We are excited to continue researching for new ideas as the latter, which would improve our prototypes and, in the end, learning about how to affront different difficulties in engineering, as well as learning about different approaches that other people may have encountered.

To finish with, we encourage students to continue with this project themselves, because knowledge is for everyone, and we feel this work has value as long as someone can learn from it.

Acknowledgments

Throughout this project we have been helped by lots of amazing people and we want to acknowledge their time, passion, and dedication to help with our issues and needs. Without them this project would not have been as it is and we want to remark what lesson each of one taught us and their importance on this project.

To start with, we want to thank Bernat Gascón and Gerard Solanes, who taught the team Arduino during High School providing the first contact with it. Furthermore, they gave our first Python Programming Classes, giving the knowledge needed for this project.

Secondly, we want to acknowledge the help provided by Edward Moriarty from the Edgerton Centre in MIT, who with his passion, knowledge and dedication taught the team some precious advice about engineering and who mentored this project from the beginning. He also gave us the opportunity to meet new people from all around the world and connect with them to learn about their projects and solutions.

Also, remark the help received by Jordi Vidal, Joel's uncle, who helped with the base knowledge of the coding part and gave mentorship to the team about how to affront some difficulties during the learning process.

Next, we want to appreciate Arnau Ortega's help, who was a mentor during the Edgerton Center workshop from MIT hosted in CIC Batxillerats and gave advice with our first prototype. Providing knowledge about Arduino and power supplies.

We also want to thank Tanner J Packham, who mentored this project during its first months with great advice about prototyping and who gave some insight into his projects and how he affronted them.

Also remark the huge help provided by Lucas K De Bonet, an MIT freshman student who has an astonishing project with a sensor glove with which we made our first shake of hands

overseas. His work and insight gave us countless help with powering our servos and giving bright advice.

Last but not least, we want to talk about Alicia Lopez, who helped us finding interesting people and to make connections with the MIT. Without her this would have been a much harder project and we are pleased to have her as a mentor.

All this help means a lot to us, and we are proud of being able to have this opportunity of working with these amazing people. We are extremely pleased with everyone, and we wanted to take a moment to appreciate their time and effort.

Thanks to everyone.

Table of Figures

Information sources

0.4.1 Online documents

[en línia] https://biomech.media.mit.edu

[en línia] https://www.cnet.com/news/how-5g-aims-to-end-network-latency-response-time/

[en línia] https://sci-hub.mksa.top/10.12968/hmed.2016.77.3.C34

[en línia] https://sci-hub.mksa.top/10.1016/j.jhsa.2016.11.025

[en línia] https://teachmeanatomy.info/the-basics/anatomical-terminology/terms-of-movement/

[en línia] https://www.machinedesign.com/markets/medical/article/21831782/whats-the-difference-between-abduction-and-adduction-biomechanics

[en línia] https://www.bbc.co.uk/bitesize/guides/zxc34j6/revision/5

[en línia] https://languages.oup.com/google-dictionary-en

[en línia] https://pubmed.ncbi.nlm.nih.gov/12888941/

[en línia] https://www.kenhub.com/en/library/anatomy/elbow-joint

(Coding Part)

[HTTP Overview] https://developer.mozilla.org/en-US/docs/Web/HTTP/Overview

[HTTP Methods] https://developer.mozilla.org/en-US/docs/Web/HTTP/Methods

[API] https://www.redhat.com/en/topics/api/what-are-application-programming-interfaces#:~:text=An%20API%20is%20a%20set,know%20how%20they're%20implemented.

[Angular] https://angular.io/

[React] https://reactjs.org/

[Frontend Frameworks] https://www.youtube.com/watch?v=cuHDQhDhvPE

[SPA vs MPA] https://asperbrothers.com/blog/spa-vs-mpa/

[Analog Pins] https://www.arduino.cc/en/Reference/AnalogPins

[Digital Pins] https://www.arduino.cc/en/Tutorial/Foundations/DigitalPins

[Rest API] https://www.redhat.com/en/topics/api/what-is-a-rest-api

[GraphQL] https://graphql.org/learn/serving-over-http/

[Session ID vs Token Auth.] https://www.youtube.com/watch?v=UBUNrFtufWo

[Stack Overflow 2021 Survey] https://insights.stackoverflow.com/survey/2021

[CORS] https://developer.mozilla.org/en-US/docs/Web/HTTP/CORS

0.4.2 Books

Knudson, D.: «Fundamentals of Biomechanics». *Springer,* (2007), p.53-56, 60-63, 82-85.

0.4.3 References

[1] Freitas, S. R., Mendes, B., Le Sant, G., Andrade, R. J., Nordez, A., & Milanovic, Z. (2017). Can chronic stretching change the muscle-tendon mechanical properties? A review. Scandinavian Journal of Medicine & Science in Sports, 28(3), 794–806.

[2] Romero, Norma Beatriz (2013). [Handbook of Clinical Neurology] Pediatric Neurology Part III Volume 113 || Main steps of skeletal muscle development in the human. , (), 1299–1310.

[3] Maw, Jonathan; Wong, Kai Yuen; Gillespie, Patrick (2016). Hand anatomy. British Journal of Hospital Medicine

[4] Shaking Hands Overseas GitHub Page: (https://github.com/Shaking-Hands-Overseas)

[5] Parvini, F., & Shahabi, C. (2007). An algorithmic approach for static and dynamic gesture recognition utilising mechanical and biomechanical characteristics. International Journal of Bioinformatics Research and Applications, 3(1), 4.

[6] Martin, Silvia; Sanchez, Eugenia (2013). Anatomy and Biomechanics of the Elbow Joint. Seminars in Musculoskeletal Radiology

[7] Karbach, Lauren E.; Elfar, John (2017). Elbow Instability: Anatomy, Biomechanics, Diagnostic Maneuvers, and Testing. The Journal of Hand Surgery

[8] J.N.A.L. Leijnse; J.J. Kalker (1995). A two-dimensional kinematic model of the lumbrical in the human finger., 28(3), 237–249.

[9] Check Theories and control models and motor learning: Clinical applications in neurorehabilitation. Neurología (English Edition), 30(1), 32–41.doi:10.1016 for more information

[10] https://graphql.org/learn/serving-over-http/

[11] https://insights.stackoverflow.com/survey/2021

[12] https://developer.mozilla.org/en-US/docs/Web/HTTP/Methods

[13] https://insights.stackoverflow.com/survey/2021

[14] Biryukova, E. V., & Yourovskaya, V. Z. (1994). A Model of Human Hand Dynamics. Advances in the Biomechanics of the Hand and Wrist, 107–122.

[15] Fok, K. S., & Chou, S. M. (2010). Development of a finger biomechanical model and its considerations. Journal of Biomechanics, 43(4), 701–713.

[16] Valero-Cuevas, F. J. (2005). An integrative approach to the biomechanical function and neuromuscular control of the fingers. Journal of Biomechanics, 38(4), 673–684.

[17] L., J., Perez-Gonzalez, A., C., M., E., B., Vergara, M., L., J., … Morales, A. (2011). Towards a Realistic and Self-Contained Biomechanical Model of the Hand. Theoretical Biomechanics.

[18] Cerveri, P., De Momi, E., Lopomo, N., Baud-Bovy, G., Barros, R. M. L., & Ferrigno, G. (2007). Finger Kinematic Modeling and Real-Time Hand Motion Estimation. Annals of Biomedical Engineering, 35(11), 1989–2002.

[19] Valero-Cuevas, F. J. (2009). A Mathematical Approach to the Mechanical Capabilities of Limbs and Fingers. Progress in Motor Control, 619–633.

[20] Parvini, F., & Shahabi, C. (2007). An algorithmic approach for static and dynamic gesture recognition utilising mechanical and biomechanical characteristics. International Journal of Bioinformatics Research and Applications, 3(1), 4.

[21] Light, C. ., & Chappell, P. . (2000). Development of a lightweight and adaptable multiple-axis hand prosthesis. Medical Engineering & Physics, 22(10), 679–684.

[22] Cuellar, J. S., Plettenburg, D., Zadpoor, A. A., Breedveld, P., & Smit, G. (2020). Design of a 3D-printed hand prosthesis featuring articulated bio-inspired fingers. Proceedings of the Institution of Mechanical Engineers, Part H: Journal of Engineering in Medicine, 095441192098088.

[23] An, K. N., Chao, E. Y., Cooney, W. P., & Linscheid, R. L. (1979). Normative model of human hand for biomechanical analysis. Journal of Biomechanics, 12(10), 775–788.

[24] https://dashboard.heroku.com/

[25] https://github.com/Shaking-Hands-Overseas/API

[26] https://github.com/Shaking-Hands-Overseas/Arduino-Hand-Firmware

[27] https://github.com/Shaking-Hands-Overseas/SHA-Driver

[28] https://hackaday.io/project/178243-lucidgloves-vr-haptic-gloves-on-a-budget

[29]https://www.pubnub.com/blog/how-fast-is-realtime-human-perception-and-technology/