Original requirements called for a **S**elective **C**ompliance **A**ssembly **R**obot **A**rm arm capable of ¼" accuracy over a 10" straight line with a drawing speed of at least 4 inches per second. Additionally, the arm should mount different drawing tools such as crayons, pencils, pens. Control of the arm is done through a Python GUI that translates simple numerical and button inputs to G-Code to be processed by the control system. The goal was to automate image drawing using a simple and manageable platform without any additional hardware or software.

We approached the design by dividing into specializations: one person per mechanical, electrical, and software role across a team of three, with the understanding that specialization would allow each person to work effectively. Breakdown of the specializations is as follows: Angel on mechanical, Caleb on software, and Lucas on electrical; work would be done independently and then integrated together.

During the development phase, we determined the two additional customer requirements: moving the tool vertically to avoid extraneous marks, and adding a multi-tool holder to automate tool switching. With seven requirements, we determined general details such as the embedded processor (a Raspberry Pi), programming language (Python), general arm and tool grasper structure; group discussion evaluated the effectiveness of certain ideas and revised or discarded them.

During the prototyping and production phase, each worked to manufacture their blocks on time. Mechanical systems used Autodesk's Inventor software in conjunction with Ender 3 3D printers to rapidly iterate through component designs. Electrical design started at the schematic level before moving to KiCAD 6.0 and used OSH Park for PCB manufacturing. Software implemented simple low-level functions such as step() or toolOpen() and integrated them into high-level functions such as drawLine(), moveToPoint(), or retract().

The revision phase was not as in depth as we wanted it to be. External restrictions on time meant few thorough design reviews. We were all able to work efficiently *independently*, but this excessive independence came at the cost of time lost since team members were not always checking in with each other and verifying delivery of blocks. Issues such as thin PCB traces, overheating motors with PLA deformation, software copy-paste bugs, weren't discovered until final integration.

Ultimately, we were unable to implement a functioning system.

Electrical issues delayed software integration which delayed control algorithms debugging. But once the system moved, mechanical bugs were then discovered and introduced delay. Revisions made only intensified this circular series delay: delays would delay discovery of issues which would cause more delays.

In this team's reflection, we consider our lack of cross-checking and regular meetings to be the main source of failure; the addition of which would've maintained the strict deadlines defined in our timeline and picked up on simple and complex errors. We did not consider the technical issues to be a major challenge; these were all solvable, but rather our management and accountability the major bottleneck.