Software Development Life Cycle (SLDC) is a process that describes the various occurring in the development of a software project. A typical Software Development Life Cycle consists of the phases below:

* A phase known as the planning and Requirement Analysis is the most important stage in SDLC as it requires the more tenured members of a software development team to work alongside the customer to develop a plan of attack to approach the project at hand.
* Defining Requirements is the stage where we document our requirements from the previous stage and make them more formal. Risk factors are also weighed in this step.
* Designing the product architecture consists of developing an architecture specified in SRS based on the requirement at hand. Many different approaches are possible but it is the team's job to come up with one with the lowest risk outcome. It is all documented in the DDS (Design Document Specifications)
* Building the product is the step everyone knows or sees as a team starts to write up the code per the DDS requirements and solve issues seen in implementation along the way
* Testing the product is a very crucial step since we need to make sure that the customer's product works without any hiccups or downtime. Fixing any bugs that pop up is the bulk of this step.
* Deployment occurs when the product has been heavily developed and stress tested and is deemed worthy of being released to the customer's audience. This step can often occur in stages like somewhat of a “soft launch”. The product will be monitored after release and maintenance will be regularly kept up to keep the product in the best shape possible for the customer.

These steps are very important to have a successful product. The steps are not always followed and some can be not fully completed which results in a product that has a shortened lifespan as it is missing an important aspect of its initial software design. To ensure these are being followed a waterfall model design can be used which makes sure that each step is fully completed before moving on to the next one. Regardless of which models you use, it is necessary to document each step of your product’s development along the way so that in the future if you or your fellow developers are looking at a product they don't have to waste time relearning something that could have been avoided with a few lines of documentation. An iterative design model, on the other hand, starts with a simple implementation almost like the opposite of the waterfall design and builds off of its almost evolving versions until the full system is implemented. This way they can gradually develop the product in an iterative fashion. The spiral model combines elements of both waterfall and iterative models and consists of four phases: planning, risk analysis, engineering, and evaluation. This model goes through multiple iterations, with each iteration building upon the previous one and this model is suited for large and complex projects. The V-model is a model that is depicted in the shape of a “V” which is similar to the waterfall model but adds a testing phase at each step of the development cycle. The two main sections of this model are the development section and the testing section. The Agile model uses short development cycles called sprints which last a few weeks where each developer is tasked with a list of work that needs to be accomplished during the weeks. This model emphasizes flexibility and collaboration and is used a lot in modern software development. Lastly, the RAD model delivers a product in a short time ram using pre-built components and reusable code. It is an iterative process with each iteration building upon the previous one and is best suited for when projects are in a time crunch. There are many SLDC models that are far better than each other for their own reasons and I believe that they should all be evaluated based on the needs of the project and then chosen accordingly.