# Team 4 Requirements Specification

**Members:** Shane Alvarez, Ryan Chartier, and Simeng Wang.

These are the user stories for the AutoGui application currently in development. The only role is a programmer, whom we assume knows how to program in basic Java.

***As a [user role], I want to [goal], so I can [reason].***

|  |  |  |
| --- | --- | --- |
| User Role | Goal | Reason |
| Programmer | ***Program a widget’s dimensions within the method annotation*** | ***Create a widget with the dimensions I want*** |
| Programmer | ***Designate a method as a Swing Widget*** | ***initialize a GUI containing that Widget*** |
| Programmer | ***Button’s listener information be contained within the annotated method*** | ***It responds to input through the GUI*** |
| Programmer | ***Create more than one Widget*** | ***Instantiate a more robust GUI*** |
| ~~Programmer~~ | ***~~Program the dimensions of the GUI into the class-level annotation~~*** | ***~~Initialize a GUI with those dimensions~~*** |
| Programmer | ***Designate the interactions between the annotated widget and other widgets*** | ***Functional interactivity between them inside the GUI*** |
| Programmer | ***Not have to import Java Swing into my annotated class*** | ***Take full advantage of the AutoGui features*** |
| Programmer | ***Not have Widgets be difficult to format and create*** | ***Expand the target audience of our application*** |
| Programmer | ***Not have to worry about the internal workings of the AutoGui*** | ***Focus on the design of my annotated class*** |
| Programmer | ***AutoGui call the Reflection library at most two times*** | ***Minimize the performance hit of my application*** |
| Programmer | ***See errors specific to the usage of custom annotations as Widget designators*** | ***Eclipse may not be able to catch these errors*** |
| Programmer | ***Create marker, single, and multi-value annotations*** | ***Greater flexibility in defining the attributes of my custom annotations*** |
| Programmer | ***Program the layout information using an annotation specific to layouts*** | ***Easily format my layout*** |
| Programmer | ***Be notified when I add a widget outside of my specified layout dimensions*** | ***Avoid cascading errors*** |
| Programmer | ***Add as little, or as much, information as I want into an annotation*** | ***Create simple to complex widgets*** |
| Programmer | ***Add widgets into a nested layout*** | ***Create more complex layouts*** |

## Problem Domain Model

### Textual Representation

The goal of our AutoGui is to have its main component, *Resolver*, read in annotations from an *Annotated Class* and create a GUI based on its custom annotations. *The Annotated Class* will contain everything that a standard Java class would, but it will also implement custom annotations. The *Annotations* within this class have a one-to-many relationship with the *Annotated Class*. Also, the *Annotations* have a one-to-many relationship with their *Attributes*, as one *Annotation* can house many defined *Attributes*. Both the *Annotations* and their *Attributes* cannot exist without the *Annotated Class*. Once these *Annotations* are read in, a *GUI* will be generated. A *GUI* cannot exist without a *Resolver*, which is the component that creates it.