Antenatal Iteration 1

We have 2 usecases in the project, entering a patient's information and generating a midwife's report. We are focused on that first use case at this point in the project. The frontside team was team 2 was on view and controller. The view we were planning to be very simple for iteration 1 being essentially a copy of assignment 1. We are targeting the GUI for iteration 2. The view and controller walk you through the process of adding creating or updating a visit. You can also see all visits which is useful for debugging but will not be final content.

Team 1 was on model and service duty. To save time they were directed to use the DAOs from assignment 1 and otherwise forget about the storage for this iteration. The target for the model was implementing the first handful of fields and looking into how they would implement some of the later fields. This included learning how medical fields are entered like blood pressure's ###/## and analyzing which fields might need math done on them at a later point in the project. The first few included some elements that had multiple pieces and would need to be broken up into their own class like parity. The service does what it needs to at this point and was adaptable enough to go back to what the controller ended up being.

One significant accomplishment of this iteration was the end-to-end storage and retrieval of our objects. The service and model right now are a great starting point to expand further in the next iterations. The model itself may need a bit of refactoring. Interior classes like Parity will be more useful elevated into their own file where they can be a shared resource with the immunizations team. The controller will need significant work in the future. We were planning on starting a GUI view in iteration 2 and the current form of the controller will not work well with a GUI view.

All risks at this point are new. We are not abstracting or genericizing our methods very well. We will probably run into some pitfalls if large changes are needed to be made which would ripple through the project right now. Another risk is falling behind on the view and controller portions and communication between sections of the project. We didn't accomplish the goal I set of a single object being passed to the service instead of a bunch of primitives so we could modify the object without changing the controller and service every time the model or view is changed.