SDLC Discussion

The SDLC model that was chosen for our Vacation Planner project lines up the best with our small group size and natural tendency of the group members involved. We chose to use the Agile model with a mix of phased development instead of other models. Our group decided against the Waterfall Model since it lacks iteration, and we already have a handful of optional requirements that could be added if time allows. Likewise, the V-Model would not have been a great choice since it also needs rigid requirements. We chose against the prototyping model because of our strict deadlines. The transformational model requires extremely precise functional specifications, and our group continues to come up with new requirements. The spiral model was not selected for the same reason: time consumption and changing requirements can cause difficulties.

The Agile Model makes the most sense for our project due to its small scope, its relative simplicity, and the focus on individual ability. Each of our group members will be working on a facet of production while working closely with one another to ensure smooth operation. The speedy nature of the Agile Method will also allow us to work at a steady pace while also being able to handle other jobs given by our group leader. The implementation of phased development through the agile method will allow us to produce an early model of our system that can also be easily modified. As for the general issue of customer availability, when considering the disadvantages of the Agile Methods, our customer is fictitious, and our requirements will be mostly determined by potential users which includes ourselves and family/friends. Because of this, it will be easy to communicate and collaborate with our users.

One major issue the project may suffer from is software bugs due to the lack of documentation; however, due to the small size of the program, we should be able to easily run through the code and locate any faults in the web app. Additionally, if major changes are made to the requirements, revising the system will be more difficult since agile does not account for future changes very well, particularly models similar to Extreme Programming (XP).