To construct a model that predicts the number of bugs a tester will find, I would start with some exploratory data analysis. I would check which features contain numeric values and which are categorical. I would also add columns to the existing tables, create a new table, or construct a view which combined the given data and some engineered data. Specifically, I would include the total number of bugs found per tester per device to the devices table. I would also add the total number of bugs historically found for each tester to the testers table. Since the devices table contain several versions of each device (such as iPhone4 and iPhone5), I would even consider combining several devices into more broad categories. The thought process is if a tester is particularly talented at discovering bugs for an older version, he might be well suited to find bugs for the newer versions.

Once I have all the features I want/think I need, I would create dummy variables for any categorical features that are considered for the modeling phase. I would then continue my analysis with a pair-plot via the package Seaborn. This would provide me a quick glace for each variable plotted against every other variable in a one to one comparison. This quick glance could give me a general sense if any two variables are collinear. If any two variables are collinear, I would remove one from the table. After the pair-plot I would plot the variables together. This would give me insight into whether or not the variables chosen are separable or not. The overall plot could also give me insight into whether clustering would be a good option.

Since there is previous data to help with the predictions, I would use a supervised learning approach. For the modeling, I would first label total bugs found as my dependent variable and all other features chosen as my independent variables. I would split the testing data into a training set and a testing (holdout set). Next would be to perform cross validation on my training set with the models I would consider. The models I would choose would depend on the analysis performed prior to modeling. If the data shows a general linear trend, I would choose a linear based model such as Ridge, Lasso, or Linear Support Vector Regressor. If there was non-linear separability I would consider a model such a Support Vector Regressor. If there was not any obvious separation, I would choose a tree based model such as a Random Forest Regressor. The cross-validation would help to reduce variance the metrics chosen.

The metrics I like to test for with a regression model such as the one explained are mean absolute error, mean squared error. I like to consider both metrics for different reasons. Mean squared error provides a robust solution while mean absolute error is more robust to outliers.

Once the cross-validation is finished and a model has been selected, I would use the testing set on the model and retrieve the metrics. I would then tune any hyper parameters the model might contain via a grid search.

Finally, I would put together a simple app. This should help any app developer that will be making a fully functional application get a sense for how new information will be inputted and displayed.