Group 19

Project 4 Integration Strategy

06 November 2021

Integration Strategy

When designing project 4 our group used the approach of unit testing each module separately before beginning integration. Our group had to adopt this strategy when we worked on the project separately. Some downsides to this include general bugs being harder to resolve. We overcame this by thoroughly testing any newly implemented module. An example of this is the integration of a new controller. In this example we were missing the UI to link with the controller, so we were unable to test the logic in the controller. This is an example of an all-at-once integration strategy, and we based most of our integration on this method because it expedited our production time. Doing the all-at-once strategy allowed us to quickly assess any group members code and perform bug fixes on the module at large. However, we did not entirely opt for a speedy integration strategy because of the major flaws when it comes to the vagueness of an error as well as the risks associated with catching a massive flaw too late into the project. To circumnavigate these issues, we incorporated the top-down method when integrating the goals class. This method of integration came in handy because of the dependencies the goals class has on the transactions class. In other words, the user input for transactions directly affects the user’s goals. After fully implementing and integrating transactions, we were able to easily incorporate the functionality of adding goals. Our group used a similar integration strategy when adding the graph features on the insights section of our budget tracker. Top-down integration allowed us to pinpoint problem areas in our goals and insights page since these functionalities depended on user transactions.