Input Validity

The input validation implemented contributes to the correctness of our app. If there are issues detected with the user's input in certain fields after they click on the Generate button, a window dialogue will be displayed informing the user of what they need to correct and an .ics file will not be created until they do. The following are some of the input validations done for our app:

1. Date/Time: The user is prevented from selecting a start date/time prior to the current date/time, from selecting an end date/time prior to the selected start one, and required to select both a start and end date/time.
2. Recurrence: The user is prevented from selecting an end date/time prior to the selected start one, from selecting exception dates/times that are after the selected end date/time and that are before the selected start date/time, and is required to select a frequency that is at least one. If the frequency is set as none, then none of the other options will be entered into the ics file, since they are dependent on having a recurring event.
3. Address (Geolocation): The user is prevented from inputting an invalid physical address. In addition, the Zip Code can contain only numbers and the City cannot contain any numbers.
4. Non-null fields: The user is required to fill in the Event Title, Priority, Start/End, Venue, and Address fields. If the user were to choose to fill in certain non-optional fields, then checks are implemented to ensure that the user supplies the accompanying input fields. For example, if the RSVP checkbox is checked, then the user must fill in the RSVP email address field. If certain Recurrence fields are filled in, then as stated in 2., certain accompanying fields must be filled in with valid input.

Testing our App

We tested our app by generating .ics files each time we wanted to confirm that correctly formatted input were being written to the .ics files, especially after a significant update to the code. In addition to testing with correct input, we also tested how incorrect input was handled in our app and that the appropriate alert dialogues would appear and that no .ics files would be generated until incorrect input is fixed by the user. The implemented input validations mentioned above can also be viewed as what we specifically tested for and did afterwards to remedy the issues we found. We also used Google Calendar afterwards to actually be sure that the correctly formatted .ics file would also be correctly displayed/read in an actual calendar app.

Usefulness

In addition to the implemented input validations, our event planner is useful also because of how the user interface is relatively simple to use. For example, the more descriptive and self-explanatory labels were used for the input fields rather than their RFC component names: 1) Resources → Needed items, 2) Classification → Visibility, 3) Location → Venue, and 4) Geolocation → Address.