QA automation Challenge

We are requesting that you produce a test automation project to ensure the quality of an app. The purpose of the automation project is to reduce or eliminate the need for manual regression tests of this app. Expect the automation to be ran on multiple iterations of the app, does not need to account for installation of the app. There may or may not be bugs in the version of the app supplied to you, though your project should uncover existing bugs and protect against future regression issues. The source code as well as a debuggable .apk(.app for iOS) have been provided. Use whichever method to produce automated tests you are most comfortable with. Please provide a functional test suite as well as instructions for running your tests.

1. https://github.com/inmotionsoftware/qa-automation-test
2. The automation project needs to uncover bugs and defects in the app that do not conform to the app requirements listed above
3. The UI and the API need to have tests that thoroughly test their behavior
4. The project should produce meaningful logs or reports that reasonably convey the bugs in the app. An individual with familiarity to the automation project should, in most circumstances, be able to easily understand the steps to reproduce any bug uncovered by the automation tests

The app supplied to you consists of three screens that are detailed below:

Login Screen

1. Login screen has a title bar, password field, username field, and a LOGIN button
2. Screen should be titled “Inmotion QA bugger”
3. The Username must be an email address with standard format - alphanumeric string, ‘@’, Domain(gmail.com) between 6 and 50 characters
4. Password field should accept an alphanumeric string between 6 and 50 characters
5. The login button needs to be greyed out until the Username and Password field are populated with valid strings
6. The Login button should automatically switch from grey to red the moment valid data is entered into each field(pwrd & u-nam)
7. The back button should navigate user back to Cities

Cities

1. Screen should be titled “Cities”
2. User should be navigated to cities upon successful login
3. Cities should contain title bar that includes title, filter, and sort
4. A list of relevant cities should be displayed on the screen
5. The sort function will sort alphabetically on first click of sort button

6. The second click of the sort button should remove the alphabetical ordering and return cities to their original state
7. The filter function will filter by the state the cities are located within will be the only cities displayed
8. Only one filter may be applied at a time
9. In order to clear the filter select “none” from the ddl
10. The back button will navigate to the

Details

1. Screen should be titled “Details”
2. Screen should contain city name, population, wikipedia link, and a description
3. City name, population, and the description should display text
4. Population value should have standard formatting(i.e. ###,###,###)
5. City name value should have standard formatting(i.e. *City, State*)
6. The wikipedia link should be an active blue hyperlink, when it clicked it should open the link in the default browser
7. The back button should navigate user back to Cities

There is also an API portion of this challenge. The API is read only.

Cities

The Cities.json endpoint provides an array of cities, that contains the name of the city, the state the city is in, and information about how to access details about the city.

https://raw.githubusercontent.com/inmotionsoftware/qa-automation-test/master/cities.json

```
{
  "cities": [
    {
      "name": "Austin",
      "state": "TX",
      "details": "/austin.json"
    },
    {
      "name": "Houston",
      "state": "TX",
      "details": "/houston.json"
    },
    {
      "name": "Dallas",
      "state": "TX",
      "details": "/dallas.json"
    },
    {
      "name": "New York",
      "state": "NY",
      "details": "/new_york.json"
    },
    {
      "name": "Seattle",
      "state": "WA",
      "details": "/seattle.json"
    }
  ]
}
```

City Detail Endpoints

The city detail endpoint provides detailed information for each city from the /cities.json endpoint. Houston is provided below as an example and a similar response can be expected for each city in the list, but the only /houston.json needs to have tests developed.

https://raw.githubusercontent.com/inmotionsoftware/qa-automation-test/master/houston.json

```
{
  "name": "Houston",
  "state": "TX",
  "description": "Houston is a large metropolis in Texas, extending to Galveston Bay. It’s closely
linked with the Space Center Houston, the coastal visitor center at NASA’s astronaut training
and flight control complex. The city’s relatively compact Downtown includes the Theater District,
home to the renowned Houston Grand Opera, and the Historic District, with 19th-century
architecture and upscale restaurants.",
  "population": 2303000,
  "url": "https://en.wikipedia.org/wiki/Houston",
  "coordinates": {
    "latitude": "29.7604 N",
    "longitude": "95.3698 W"
  }
}
```