Home Challenge

Thank you for taking the time to do our Home Challenge. It consists of three parts:
First part is about performing exploratory testing on one of the money management applications. Second and third parts are about writing a set of automated checks for one of the Android open source applications and for a sample REST API.

First task in the **Home Challenge** is mandatory. You can choose between second and third task according to your experience. Completing all three tasks is a plus.

Please submit your results by sending us a link to your github repository named {firstname}-{lastname} containing:

- A single **markdown file** with the exploratory testing charters
- Separate folder containing test automation implementation

Tasks

1. As **Software Quality Engineer** you are given a task of performing exploratory testing on one of **Monefy** mobile applications. Choose the platform your are more
   familiar with (iOS or Android) You can download the apps here:

    - iOS
    - Android

Monefy is a money management app where you can add your expenses in a dedicated category and track your spendings. You can review your expenses in detailed charts.

Task

Please write down:

- Exploratory testing charters to document your testing. You can use simple format of:
    - Explore *target*, with *resources*, to discover *information*

- Findings from your charters. Did everything work as expected? What bugs were discovered?
- Prioritisation of those charters - which area of the app or testing would you explore first and why?
- How much time you have planned for each charter?
- What kind of risks you need to mitigate for this type of application?

> *We want to see how you approach and organise exploratory testing and prioritise your work. Do not spend more than 2-3 hours on this task.*

---

2. As **Software Quality Engineer** you are given a task of writing a set of automated tests for `Monefy` Android application used in Task 1.
   You can download the APK file here.

Task

- A list of ideas/bullet points you would test for. Focus on good coverage, rather than complete test cases.
- A prioritization of such test cases according to their possible business impact in case of failure.
- Outline the possibilities of automating proposed test cases on different levels, together with a short summary of pros and cons of each of them.
- Please automate proposed test cases.
- A short explanation of the provided solution inside the README file.

Rules

- Feel free to use whatever frameworks, library, packages you like
- Use any familiar structure or pattern you are comfortable with
- Include README about how to use the test suite. Take care about explaining the setup, as we might not have the same libraries installed.
- Please explain your approach and why you chose the particular tech stack

---

3. As **Software Quality Engineer** you are given a task of testing RESTful API in Best Buy API playground. API playground is a training pool designed to learn about an API framework in local environment. Set up your environment as explained in the guide

Task

- Set up the API playground and run it locally.
  If you face issues with node 11 not being compatible with sqlite3, use the commands below to install node 8.

```
brew unlink node

brew install node@8

brew link node@8
```

- A list of test cases proposed for automation.
- Please automate proposed test cases.
- A short explanation of the provided solution.
- **Note:** Make sure your test automation solution is **NOT a Postman suite** but a **Framework** built using best coding / design practices.

Rules

- Feel free to use whatever frameworks, library, packages you like
- Use any familiar structure or pattern you are comfortable with
- Include README about how to use the test suite. Take care about explaining the setup, as we might not have the same library installed.
- Please explain your approach and why you chose the particular tech stack

Tech at N26

N26 is completely hosted in the cloud. Our tech stack includes Kotlin, Java, docker, Jenkins, Nomad, Consul, Vault, Saltstack, Terraform, Spring boot, Kafka, Kubernetes, AWS, Appium, Cypress, nodeJS etc. These tools and technologies are fully embraced by our test frameworks. Bonus points if you recognise most of them.

Let us know if you have any questions.