Visualizing GitHub Flow

In this tutorial we will illustrate GitHub Flow using the visualisation tool:
https://git-school.github.io/visualizing-git/#free-remote
The tool is simply a visualisation, a simulation, of git commands, without the actual repositories.

Some examples of using the tool can be found here:
https://github.com/git-school/visualizing-git/blob/gh-pages/README.md

(Unfortunately, creating a pull request is not a git command but a function provided by e.g. GitHub. Therefore, in this example when the remote feature branch is ready to be merged into the remote main branch it is shown as a push and not a pull request as it should be - this is mentioned in the steps below.)

1. Create a starter project by opening: https://git-school.github.io/visualizing-git/#free-remote

2. The first step in developing a new feature using the GitHub Flow process is to ensure that your local repository is up to date with remote (e.g. update in IntelliJ). To start with in the project, the local is up to date with the remote. To simulate a more realistic case where the local repo is behind the remote, we'll make a commit to the remote. Change to the remote in the visualisation tool by entering:
   ```
   mode remote
   ```
   Now enter the git command:
   ```
   git commit -am 'welcome'
   ```

3. Change to the local repo in the visualisation tool by entering:
   ```
   mode local
   ```

4. To synch the local with the remote, enter the git command:
   ```
   git pull origin master
   ```

5. The local is now in synch with the remote and we can create a local feature branch from main. Enter the git command:
   ```
   git checkout -b feature-login
   ```

6. Make a few commits on the feature-login branch. Enter the git commands:
   ```
   git commit -am 'added login'
   git commit -am 'added remain logged in'
   ```

7. While we’ve been working on the login feature in our local repo, our colleagues have been busy and made commits to the main branch on the remote repo. We’ll simulate this by changing to the remote in the visualisation tool and make a few commits. Enter:
   ```
   mode remote
   ```
   Now enter the git commands:
   ```
   git commit -am ‘profile name’
   git commit -am ‘profile image’
   ```

8. We’ve completed the login feature in our local repo and would like to merge our local login feature branch in to the remote main branch. We should first check that no merge conflicts will arise by:
   1. Synching (updating) our local main branch from the remote repo (and resolving any merge conflicts).
      Change to the local repo in the visualisation tool:
      ```
      mode local
      ```
      Enter the git commands:
      ```
      git checkout master
      git pull origin master
      ```
   2. Merging the updated local main branch into our local feature branch (and resolving any merge conflicts.)
      Change to the feature-login branch:
      ```
      git checkout feature-login
      ```
      To merge the master branch in to the feature-login branch enter:
      ```
      git merge master
      ```
      Any merge conflicts should be resolved.

9. The local feature-login branched can now be pushed to the feature-login branch on the remote. (The feature-login branch on the remote will be created.) Enter the git command:
   ```
   git push origin feature-login
   ```

10. The remote feature-login branch is now ready to be merged into the remote main branch. This would be achieved by creating a **pull request** on GitHub. (This function is not available in the visualisation tool - we’ll merge instead for completion here.)
    Change to the remote in the visualisation tool:
    ```
    mode remote
    ```
    Checkout the main branch:
    ```
    git checkout master
    ```
    Merge the feature-login branch into the main branch:
    ```
    git merge feature-login
    ```

11. In practice, the pull request would be reviewed, any required changes implemented and then merged. The login feature branch can then be deleted.

Here are all the visualization tools commands used:

```
1. mode remote
2. git commit -am ‘welcome’
3. mode local
4. git pull origin master
5. git checkout -b feature-login
6. git commit -am 'added login’
7. git commit -am 'added remain logged in’
8. mode remote
9. git commit -am ‘profile name’
10.git commit -am ‘profile image’
11.mode local
12.Git checkout master
13.git pull origin master
14.git checkout feature-login
15.Git merge master
16.git push origin feature-login
17.mode remote
18.git checkout master
19.git merge feature-login
```