German University in Cairo
Faculty of Media Engineering and Technology
Prof. Slim Abdennadher

CSEN 704/DMET 706, Advanced Computer/Media Lab
Winter 2018

Project; Proposal Guidelines

Submission Week: **13/11/2018**

The aim of the project is to build a web application that complies with The Twelve-Factor App methodology. This document describes the main guidelines by which the submitted project proposal should abide. The general project theme is "**Services**". All proposed projects should lie under this theme, and should answer the following questions. There is no restriction on the programming language(s) used.

1. **What is your application called?**

   Pick a unique and descriptive name. It's how your application will be referenced from now on.
   Ex.: GUC Pal.

2. **What does your application do?**
   What is the functionality provided by your application? How will it impact the world?
   Ex.: GUC Pal helps CSEN704/DMET706 students schedule office hours with their TAs.

3. **Which backing/3rd party services will your application use?**
   Minimum: 2. Which backing and/or 3rd party services will your application collect its data from and forward it to?
   Ex.: GUC API and PostgreSQL Database.

4. **How will your application use the backing/3rd party services?**
   Which features of the backing and/or 3rd party services will your application access? How will it benefit from the them?

   Ex.: The GUC API will be queried to get the TAs' free slots. The PostgreSQL will be used to save the timings of the office hours.

Requirements

Your application **should** comply with the below selected factors

0. **README**

   - Your repository should contain a README file that has instructions on how to run the application. More details follow.

1. **I Codebase**

   - Any code related to your application should be pushed to a **private** Git repository provided by Github, Gitlab, or any similar service.
   - The code should be pushed via multiple commits, from different collaborators, at different timings, to show each member's progress and contribution to the project. Excuses like "we all worked on the same machine using the same account" will not be accepted.

2. II Dependencies

- Code dependencies should be declared using a **dependency declaration tool** and isolated using a **dependency isolation tool** (e.g. govendor for Go, npm for NodeJS, bundler for Ruby…etc). Instructions on how to run these tools should be included in a README file.

3. III Config

- Application configuration that vary across deploys should not be hard-coded.
- You're free to use either a separate **config file** that is not a part of the repo (remember to include an example), or using **environment variables** (with instructions on how to use them clearly stated in a README file).

4. V Build, Run, Release

- Your repository is expected to contain one or more Dockerfile(s) that can be used to run off your application in one `docker run` command. Any required flags should be clearly documented in a README file.

5. IV Backing Services

- Your application should rely on **at least 2** backing services.
- Your repository is expected to contain a docker-compose.yml file that can be used to bring up your application, along with any dependent backing services (those being actual services or mocked ones) in one `docker-compose up` command. Any required flags should be clearly documented in a README file.

Bonus

Including one or more of the below optional requirements might grant you extra marks:

- Deploying your application on a server using Docker and/or docker-compose. Surprise me with a working deployment on Kubernetes (and its deployment.yml) and you'll get on my good side ☺.
- Having an extra-ordinary, out-of-this-world frontend interface.
- Complying with all 12 factors in a sensible manner, i.e. in ways that actually do make sense, and not by just adding the bare minimum to meet the factors' rules.

Submission

https://goo.gl/forms/1VB0YAlSjvjEQNjo2