**Technical Test (Part 2)**

The goal of this exercise is to build a fully functional Flask App using best practices and reusable code including unit testing.

Rules

You are free to use any libraries you like, unless explicitly mentioned below.

Any dependencies should be installable via pip (except for redis). You should list them in a requirements.txt file.

Use Flask's built-in development server and the sqllite on sqlalchemy database.

Assume redis-server is installed on the test machine with the guest account available.

The Task

Part 1. Initial implementation

* Model a Contact with username, email, first name and surname.
* Create a Restful API that returns a list of all contacts.
* Returns a contact by username.
* Saves a Contact.
* Updates a Contact
* Deletes a Contact

Unit Tests should be included.

Part 2. Extension

Introduce a new Email model and refactor the email field into the new table.

- Allow a contact to have multiple email addresses. Adjust GET, POST, PUT, DEL methods to the new sub entity.

- Extend the GET to also accept the email address for contact retrieval.

Unit Tests should be included.

Part 3. Celery

Introduce a celery task to create a random contact with two email addresses every 15 seconds. Any older entries then 1 min should be cleaned up and deleted.