# Homework for all the DevOps

You will build a sample RESTful application that will be evaluated and referenced during the interview process. The application is an event management application (i.e. Evite) with a RESTful API and the requirements for the app are listed in the Critical Deliverables section below.  Should you have additional time and you choose to you can work on and submit the Nice to Have deliverables defined below.  The interview process will focus on the Critical deliverables but, if you submit Nice to Have deliverables, please be prepared to talk about those topics as well.  In order to give us time to evaluate the application and prepare for the interview please submit the application no later than 4 days. If you have questions on any of the requirements below please feel free to reach out for clarification.

Requirements

- The tool should be submitted with a pre-defined events in the data store. Events should have the following fields identified:
- An event will have a name
- An event will have a location
- An event will have a start and end time
- An event will have a unique identifier
- The tool should accept an email address as a unique identification for a user (when signing up for an event).
- The tool should allow the user to
- List all events
- Sign up for an event
- Remove email address from event
- When signing up for an event the tool should email a pre-defined email address with a notification.
- All properties (i.e. the pre-defined email address) should be easy to change before deployment.
- All event times will be in the same timezone.
- An event can span multiple days.

Deliverables

Critical

- The code should be build and deploy locally (no dependencies on AWS, Heroku, or other external services)
- There should be documentation on how to start the application
- All dependencies should be crafted in their associated package managers (Gemfile, requirements.txt, etc)
- The backend code should be written in Python, Ruby, or Perl.
- The code should use SQLite as a datastore.
- The REST endpoints should be well documented for public consumption
- The provided email address can only sign up a unique event once

Nice to Have

- The code should have and pass all unit tests
- The tool should have API endpoints for managing events
- The tool should have a front end (written in JS) for signing up for events / managing events.
- The tool should email the provided email address with a calendar invitation to the event.
- The tool should have APIs (protected with an fixed API key) that allow a user to:
- See all people who signed up for an event
- Remove an email address from an event
- Sign up a person from an event