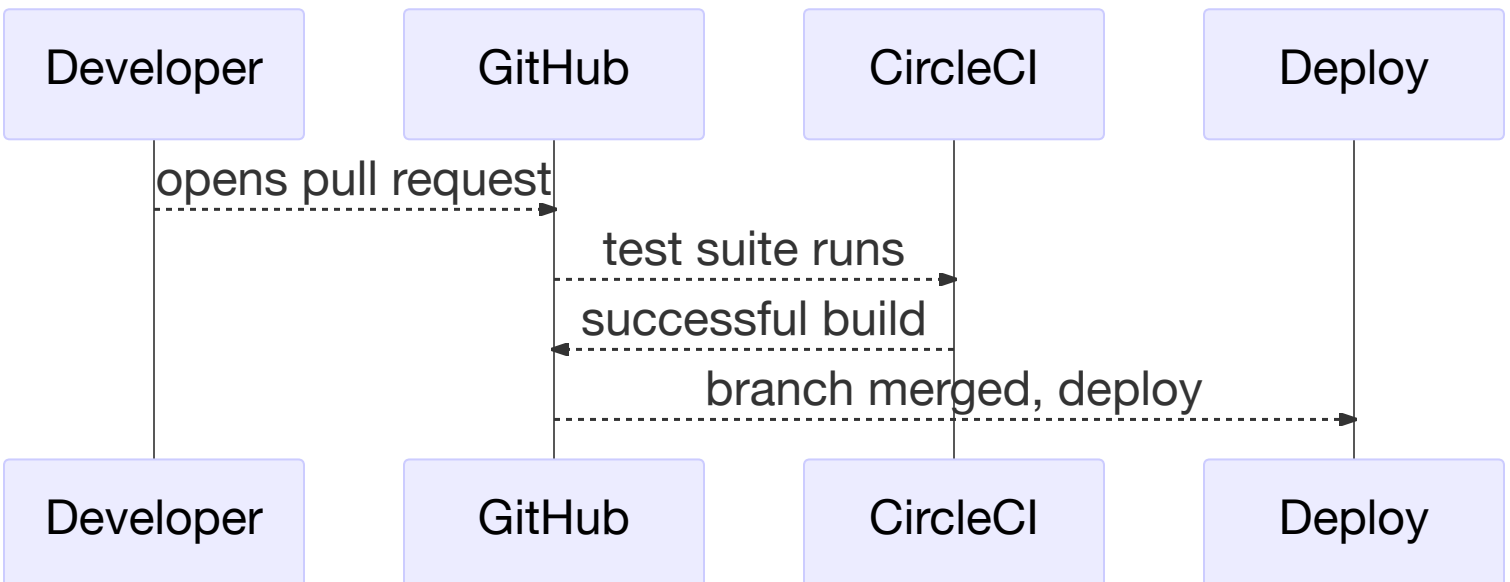
ZDCD: zero downtime continuous delivery

using django, postgres, and heroku.

What it takes to deliver features

at the speed of `git commit && git push`

workflow without ZDCD



common pitfalls

- database schema migrations
  - field alterations, field renames, or removal of fields can cause downtime while the operations run
- downtime while a user is actively using the application
- complexity overhead when adding features
- poor integration test coverage can lead to surprises

benefits of continuous delivery

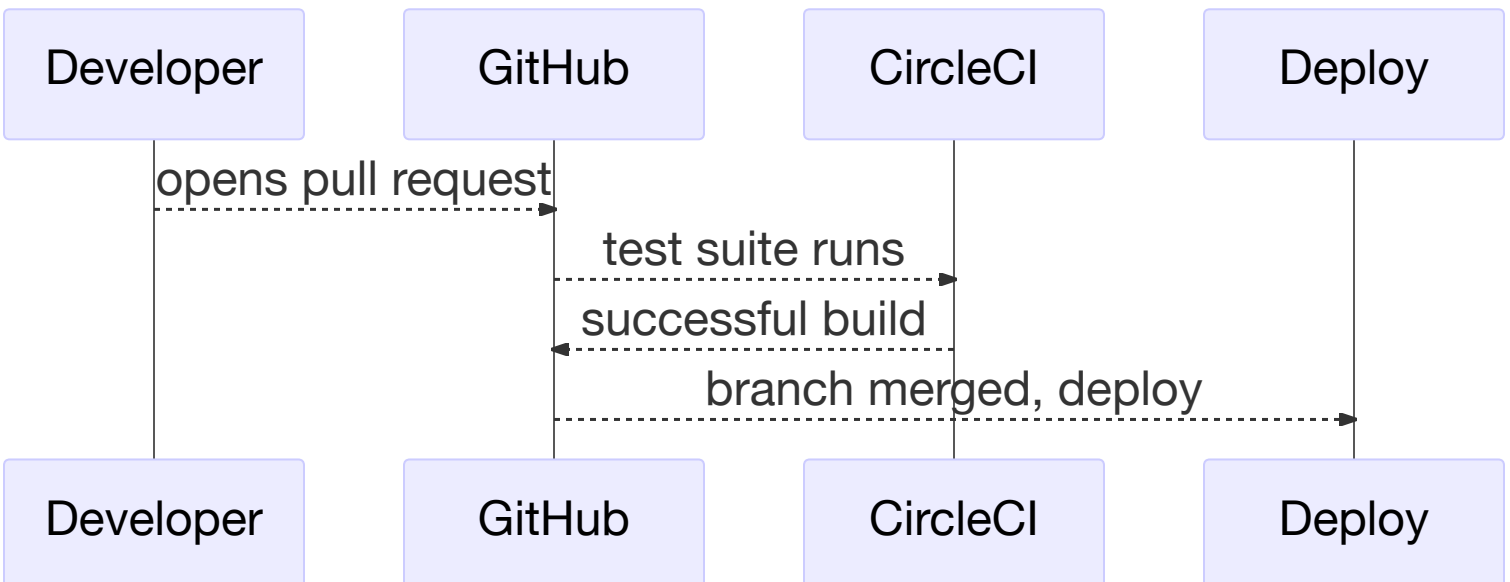
- releases to production can happen multiple times a day
- no release schedule
- feature iteration & feedback cycle will be measured in hours or days, not weeks
- product team happiness increases when user feedback is quicker
- product manager can test ideas with smaller customer groups before launching to general availability

database schema migrations

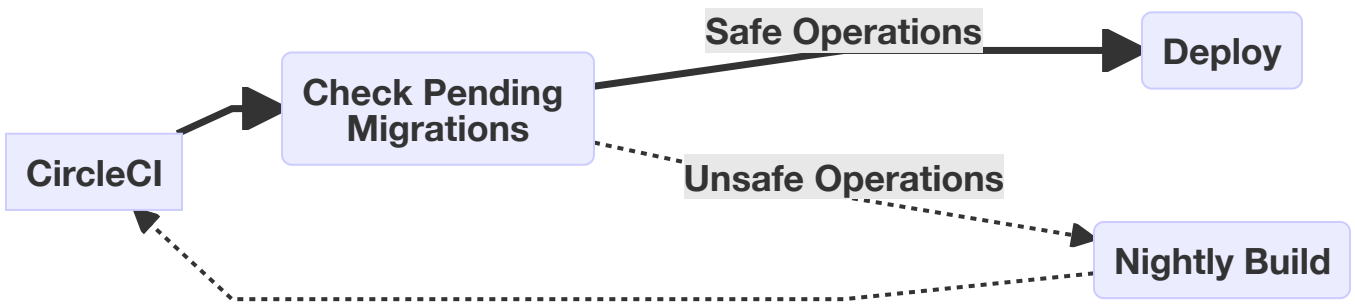
- migrations operations are either safe or unsafe. safe operations include adding a new column, with the default value set to null, or a new table. unsafe operations include renaming a column/index, changing column/index attributes, or removing a column/table/index.
- migrations deemed safe require zero downtime to apply, and will not affect application usage. applying migrations before pushing code will allow new code to operate correctly.
- migrations deemed unsafe require downtime, and can only be applied during a nightly build. nightly builds become a blocker, and builds will be queued up and applied in a single deployment.

workflow with ZDCD

workflow does not change with ZDCD!

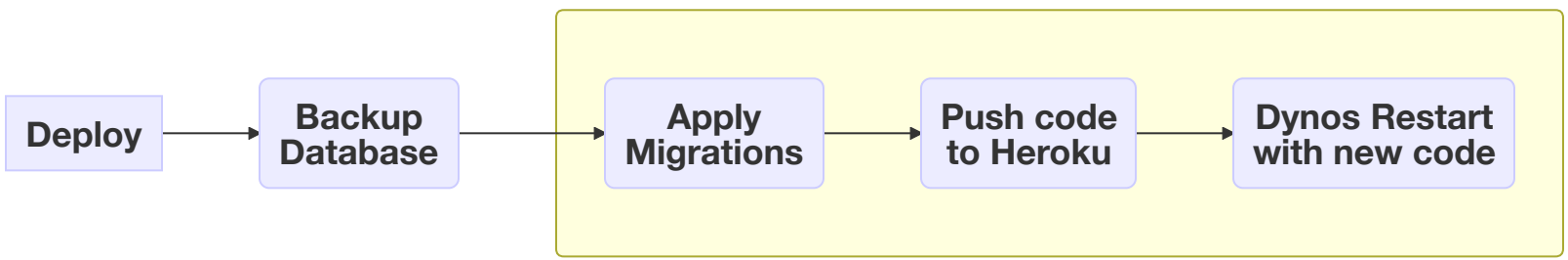


deployment preflight check



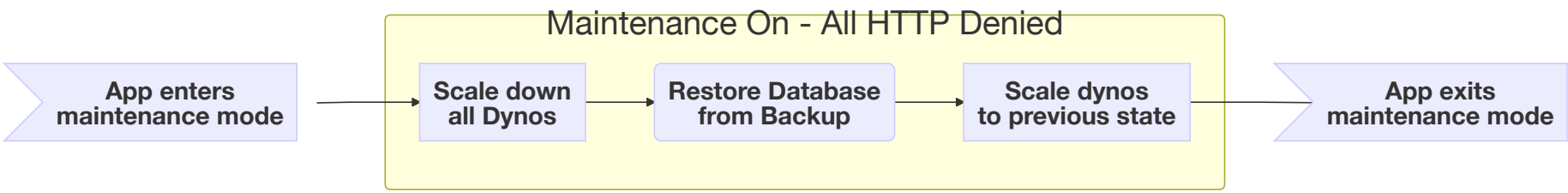
- nightly build will run every night, may not always trigger a deployment
  - *example: saturday no code is committed, so deploy is skipped*
- nightly builds skip the migrations preflight check

deployment process



- the deployment process is identical for ZDCD or nightly builds. if any of the highlighted steps fail, the deployment failure process will kick off.

deployment failure recovery



- a deployment failure will automatically rollback to the previously running database and build.
- during a rollback, downtime is required only if the database was migrated during the release process.
- if a database restore is required, then the application will be placed in to maintenance mode on heroku, effectively blocking all incoming traffic.