Iteration #1 - Plan

Features:

Required Technologies (component-view):

1. Frontend SPA hosted on ECS / Fargate
2. Backend Lambda Functions with database and codebuild interfaces
3. DynamoDB tables and object definitions
4. Codebuild resource generation / teardown with lambda (and likely cloud formation)
5. S3 persistent / temporary object storage

Front-end:

1. Homepage with clickable components
2. Student interface with only 1 practice to test

Proposed integration pipeline:

1. Frontend javascript hosted on ECS
2. Backend lambda → running builds, query relations
3. Database → loading relational data into lambda

Database:

User Table

(user_id, name, cognito_id)

Primary key → user_id
Sort key → name (debug)
Foreign key → cognito id (aws cognito interface)

Classroom Table

(classroom_id, owner, List<user_id>, List<problem>)

Primary key → classroom_id
Sort key → owner
Foreign key → owner (user_id of professor)

Problem Table

(problem_id, s3_location)

Primary key → problem_id
Foreign key → s3_location (aws s3 folder containing problem build src [input])

KEY Relations

Admin → professor
Professor → classroom
Classroom → student

Classroom → problem set (problem_id, s3_location)

Student request problem
Server returns problem
Student submits problem request

Frontend (html) → js (server) → api gateway → lambda (interface)

Lambda (output) → api gateway → js (server) → frontend (html)