UDACITY

PROJECT SPECIFICATION

FSND - Capstone

Data Modeling

CRITERIA	MEETS SPECIFICATIONS
Architect relational database models in Python	• Use of correct data types for fields • Use of primary and optional foreign key ids
Utilize SQLAlchemy to conduct database queries	• Does not use raw SQL or only where there are not SQLAlchemy equivalent expressions • Correctly formats SQLAlchemy to define models • Creates methods to serialize model data and helper methods to simplify API behavior such as insert, update and delete.

API Architecture and Testing

CRITERIA	MEETS SPECIFICATIONS
Follow RESTful principles of API development	• RESTful principles are followed throughout the project, including appropriate naming of endpoints, use of HTTP methods GET, POST, and DELETE • Routes perform CRUD operations

CRITERIA	MEETS SPECIFICATIONS

Structure endpoints to respond to four HTTP methods, including error handling	• Specifies endpoints and behavior for at least:
	  ◦ Two GET requests
	  ◦ One POST request
	  ◦ One PATCH request
	  ◦ One DELETE request
	• Utilize the `@app.errorhandler` decorator to format error responses as JSON objects for at least four different status codes

Enable Role Based Authentication and roles-based access control (RBAC) in a Flask application	• Project includes a custom @requires_auth decorator that:
	  ◦ get the Authorization header from the request
	  ◦ Decode and verify the JWT using the Auth0 secret
	  ◦ take an argument to describe the action
	  ▪ i.e. @require_auth('create:drink')
	  ◦ raise an error if:
	  ▪ the token is expired
	  ▪ the claims are invalid
	  ▪ the token is invalid
	  ▪ the JWT doesn't contain the proper action
	• Project includes at least two different roles that have distinct permissions for actions. These roles and permissions are clearly defined in the project README. Students can reference the Casting Agency Specs in the Specifications section of this rubric as an example.

Demonstrate validity of API behavior	• Includes at least one test for expected success and error behavior for each endpoint using the unittest library • Includes tests demonstrating role-based access control, at least two per role

CRITERIA	MEETS SPECIFICATIONS
	least two per role.

Third-Party Authentication

CRITERIA	MEETS SPECIFICATIONS
Configure third-party authentication systems	Auth0 is set up and running at the time of submission. All required configuration settings are included in a bash file which export: • The Auth0 Domain Name • The JWT code signing secret • The Auth0 Client ID
Configure roles-based access control (RBAC)	• Roles and permission tables are configured in Auth0. • Access of roles is limited. Includes at least two different roles with different permissions. • The JWT includes the RBAC permission claims.

Deployment

CRITERIA	MEETS SPECIFICATIONS
Application is hosted live at student provided URL	• API is hosted live via Heroku • URL is provided in project README • API can be accessed by URL and requires authentication
Includes instructions to set up authentication	• Instructions are provided in README for setting up authentication so reviewers can test endpoints at live application endpoint

Code Quality & Documentation

CRITERIA	MEETS SPECIFICATIONS
Write clear, concise and well documented code	The code adheres to the PEP 8 style guide and follows common best practices, including: • Variable and function names are clear. • Endpoints are logically named. • Code is commented appropriately. • Secrets are stored as environment variables.
Project demonstrates reliability and testability	• Application can be run with no errors and responds with the expected results. • API test suite for endpoints and RBAC behavior runs without errors or failures
Project demonstrates maintainability	• Variable names are logical, code is DRY and well-commented where code complexity makes them useful

Project includes thorough documentation	• Project includes an informative README
	  ◦ Motivation for project
	  ◦ Project dependencies, local development and hosting instructions,
	  ◦ Detailed instructions for scripts to install any project dependencies, and to run the development server.
	  ◦ Documentation of API behavior and RBAC controls

Suggestions to Make Your Project Stand Out!

- Create a frontend that works with your API - including a login that will redirect the user to Auth0. Let your work come to life on the screen!!

- Implement authorization with a tool other than email or Google. Add more options for your users' authentication flow.
- Deploy your application and database on AWS. Checkout this link and this for Postgres to get started.