Conference Organization

Description

Supporting Courses: Developing Scalable Apps with Python
Students will develop an API server to support a provided conference organization application that exists on the web as well as a native Android application. The API supports the following functionality found within the app: user authentication, user profiles, conference information and various manners in which to query the data.

Rubric

Criteria	Does Not Meet Specifications	Meets Specifications	Exceeds Specifications (Completely Udacious)
App Architecture	App is not architected as a Web Service API.	App is architected as a Web Service API.	*Not available.*
	App can not support a variety of possible front-end clients.	App supports a variety of possible front-end clients.	*Not available.*
Task 1: **Design Choices (Implementation)**	Student does not add classes for Session and SessionForm.	Student adds classes for Session and SessionForm.	Student implements additional functionality such as an entity for speakers.
Task 1: **Design Choices (Response)**	The README file does not include an explanation about how sessions and speakers are implemented.	The README file includes an explanation about how sessions and speakers are implemented.	The README file includes an explanation of the design decisions behind the additional functionality.
	Student response does not justify the implementation details.	Student response shows understanding of the process of data modeling and justifies their implementation decisions for the chosen data types.	*Not available.*
Task 2: **Session Wishlist**	Users are not able to mark sessions they are interested in and/or retrieve their own current wishlist.	Users are able to mark sessions they are interested in and retrieve their own current wishlist.	*Not available.*
Task 3: **Additional Queries**	The README file does not describe two additional query types.	The README file describes two additional query types that are consistent with the goals of the project.	*Not available.*
	Additional queries are not implemented.	Both of the proposed queries are implemented.	*Not available.*
Task 3: **Query Problem**	Student does not explain the problem with the provided query in the README file.	In the README, student describes the reason for the problem with the provided query.	*Not available.*
	Student does not propose a solution to the problem in the README file.	In the README, student proposes one or more solutions to the problematic query.	Student has implemented the proposed solution(s) in actual code.
Task 4: **Featured Speaker**	Student does not implement getFeaturedSpeaker().	Student implements getFeaturedSpeaker().	*Not available.*
	Student does not use App Engine’s Task Queue when implementing the featured speaker logic.	Student uses App Engine’s Task Queue when implementing the featured speaker logic.	*Not available.*
Code Quality	Code formatting is not ready for personal review.	Code is ready for personal review and neatly formatted.	*Not available.*
	Code does not follow an intuitive, easy-to-follow logical structure.	Code follows an intuitive, easy-to-follow logical structure.	*Not available.*

Code Readability	Comments are not present in code or existing comments do not effectively explain code segments.	Comments are present and effectively explain longer code procedures.	*Not available.*
Documentation	No README file is included.	README file is included.	*Not available.*
	README file is incomplete.	The README file provides details of all the steps required to successfully run the application.	*Not available.*