Stackoverflow Light - Backend

Functional requirements

- The app should make it possible to ask questions and allow other users to answer them
- User interface is not needed, the main focus is on the backend platform
- People can upvote or downvote questions
- The most popular questions are at the top. You can define your own definition of popularity.
- When opening the application, the user expects to see a list of questions asked. Questions should also visualize the number of answers & votes.
- Users can click on questions and see a detailed screen of the questions with the answers given by other users. You can also upvote/downvote answers
- Users can not pose questions or give answers without authenticating
- Nice to have: the ability to push real-time updates for questions, answers & votes to the client
- Allow users to view some metrics regarding:
    - Most popular day of the week
    - Average votes, questions, answers/users
    - Total questions, votes, and answers
- The ability to improve the API without breaking the client's applications when new updates are rolled out
- Any kind of application should be able to interact via a REST interface

Technical Requirements

- Provide technical architecture of the platform and a fluent way to present the created API
- Use git during development and provide us read-only access to the repository or the repositories
- You're free to choose which frameworks, libraries, databases, and/or services to engineering the solution, but we expect you to be able to defend the choices you've made.
- Make use of an OpenID connect provider for authentication
- It must be possible to deploy the application on Google Cloud, Azure or AWS. Guide us through the possible services that you will need to run your solution in the cloud.
- Ability to scale up or down based on activity and keep caching in mind.
- Use Docker and/or docker-compose to run your solution locally. You may assume an internet connection is present in order to integrate with other services.

Additional Guidelines & Tips

- The technical structure/architecture of the application is by far the most important part of this exercise. Make sure your code is clean, well-tested and well-structured.
- Don't lose too much time on details.

- Guide us through your code, the technical design choices (e.g. in a README) and the deployment process. We should be able to deploy your solution with relative ease.