	Opensource Backend in Rust	Sprint 1	Sprint 2	Sprint 3	Sprint 4
Task No.	Task				
1	Create a record through the API and UI				
2	Update a record through the API and UI				
3	Delete a record through the API and UI				
4	Read a record through the API and UI				
5	Read a list of records through the API and UI				
6	Create a collection through the API and UI				
7	Delete a collection through the API and UI				
8	Create and remove an index on a collection through the UI				
9	Create a user with username and password through API and UI				
10	Log in/out user				
11	User Logs with graphs				
12	Information website with Documentation (for the usecases completed so far)				
13	Start a Server to serve all the requests				
14	Create an Admin, Delete an Admin through the UI				
15	File Creation/Deletion/Downloading and Viewing in sdk and frontend.				
17	Embedded Documents within a document				
18	Lambda functions on database with ability to schedule them in time.				
19	Realtime Database, Frontend, Backend and SDK				
20	Stress Test Page for user to check load on server				
21	File system support with Azure.				
22	Generate Schema diagram acording to collections and their records.				
23	Implementation of design pattern singleton of the database to speed up the database and overcome any faults in the system				
24	Settings for Logs, Admin, Mail, SMTP, Azure				
25	Auth applied on the frontend and server (WIP).				
26	Logs page to view requests in real time, as well as view past requests.				
27	Bug fixes throughout the system				
28	Refactoring in server				
29	Refactoring in sdk				
30	Integrated sdk with all the latest changes and published on npm				
32	Added support for editing file and foreign key field types in database				
33	Oauth implementation using our sdk				
34	Integrated realtime feature on deployed instance				
35	Bundled frontend and server together as single executable for Linux, Windows and MacOS				
36	Role Based Access Control of the database (Involved also creating a datastructure to evaluate permissions in O(1) time)				
37	UI for logs terminal in Functions use case				
38	Check storage use and files demographic as a pie chart				