Project info

Project title: Firedata - Enabling easy cloud stats through Cloud Firestore
Project short title: Cloud Firestore Integration

URL of project idea page: https://github.com/rstats-gsoc/gsoc2018/wiki/Firedata---The-Google-Firebase-Package

Bio of Student

Student name: Jiasheng Zhu

Student affiliation

Institution: UC San Diego
Program: BS in Computer Engineering and Applied Mathematics
Stage of completion: Fourth year undergraduate, graduating in June 2018

Mentors

Mentor name: Robin Kohze Mentor email: robin@kohze.com

I been in touch with the mentor for almost weeks. We have been discussing the potential benefits and ways to improve the current packages. Robin showed a strong confidence in bringing great impacts with the new developments.

Mentor name: Samuel Schmidt
Mentor email: samuelschmidt218@gmail.com

Samuel supported the fireData development from the very beginning a year ago. With his direct involvement in the cellular data collection he is very interested in tools to share and analyse those datasets whereas fireData offers an excellent toolset.

Mentor name: Laurent Gatto Mentor email: lg390@cam.ac.uk

Laurent is a data scientist and leading the Computational Preoteomics Unit at Cambridge University. He was a GSoC mentor with the R organization for the mzR bioconductor package.

Mentor name: Bert Jehoul
Mentor email: bert@openknowledge.be

Bert works at the open knowledge organisation in Belgium and is interested in a wider adaption of the R fireData package and the integration of into openCPU web applications.

Coding Plan and Methods

This project is really about encapsulating the cool features from Firebase and make them accessible and manageable for R developers. Specific implementations include but not limited to integrating the functionalities of Cloud Firetore, enabling streaming connections and etc. I will follow the current naming convention used by Firedata and those Firebase use for other languages APIs.

This is the first time for me to work for an open-source community on a public package. My previous collaboration with Github involves pulling, fixing conflicts, committing and pushing, which should satisfy the core needs for this project. I believe I will devote some of my time to getting used to all the conventions and rules. Even though I am familiar with most tools and languages for this project, there may be places that I normally do not notice when I program solo. For example, working on a public-maintained github repository maybe different than my previous experience with Github. I will both study by myself and consult my mentor who is an experienced contributor to both the Firedata package and R community.

Adding the support for Cloud Firestore is important because it offers a cheaper alternative to realtime database and the option of querying. It enables experienced developers to find more specific methods to accomplish their tasks. For REST, there is already a total of 16 (12 documents methods and 4 index methods) supported:

Methods	
batchGet	`POST /v1beta1/{database=projects/*/databases/*}/documents:batchGet` Gets multiple documents.
beginTransaction	`POST /v1beta1/{database=projects/*/databases/*}/documents:beginTransaction` Starts a new transaction.
commit	`POST /v1beta1/{database=projects/*/databases/*}/documents:commit` Commits a transaction, while optionally updating documents.
createDocument	`POST /v1beta1/{parent=projects/*/databases/*/documents/**}/{collectionId}` Creates a new document.
delete	`DELETE /v1beta1/{name=projects/*/databases/*/documents/*/**}` Deletes a document.
get	`GET /v1beta1/{name=projects/*/databases/*/documents/*/**}` Gets a single document.
list	`GET /v1beta1/{parent=projects/*/databases/*/documents/*/**}/{collectionId}` Lists documents.
listCollectionIds	`POST /v1beta1/{parent=projects/*/databases/*/documents}:listCollectionIds` Lists all the collection IDs underneath a document.
patch	`PATCH /v1beta1/{document.name=projects/*/databases/*/documents/*/**}` Updates or inserts a document.
rollback	`POST /v1beta1/{database=projects/*/databases/*}/documents:rollback` Rolls back a transaction.
runQuery	`POST /v1beta1/{parent=projects/*/databases/*/documents}:runQuery` Runs a query.
write	`POST /v1beta1/{database=projects/*/databases/*}/documents:write` Streams batches of document updates and deletes, in order.

Methods	
create	`POST /v1beta1/{parent=projects/*/databases/*}/indexes` Creates the specified index.
delete	`DELETE /v1beta1/{name=projects/*/databases/*/indexes/*}` Deletes an index.
get	`GET /v1beta1/{name=projects/*/databases/*/indexes/*}` Gets an index.
list	`GET /v1beta1/{parent=projects/*/databases/*}/indexes` Lists the indexes that match the specified filters.

Someone with a good experience with querying can make use of the strong query features Cloud Firestore supports and run data selection and sorting on cloud (to save time and computation power). And allowing that interfaces in R becomes my mission in this project.

Especially for query construction, I should be providing a method that helps developer to construct a query that matches the API:

JSON representation

```
{
  "select": {
    object(Projection)
  },
  "from": [
    {
      object(CollectionSelector)
    }
  ],
  "where": {
    object(Filter)
  },
  "orderBy": [
    {
      object(Order)
    }
  ],
  "startAt": {
    object(Cursor)
  },
  "endAt": {
    object(Cursor)
  },
  "offset": number,
  "limit": number,
}
```

One straightforward way is to provide a method that takes in all the optional parameters and generate a JSON string that is for the user to again call the runQuery method. A more dedicated method should integrate all these functions. This Query feature is especially important because it add extra useful functionalities to the realtime databases. It can shorten the time needed at local to sort or select needed elements and instead leave the compuation to online servers.

Test

● Easy: Follow the github instructions and use the download example to get the mtcars dataset.

```
if (!require("devtools")) install.packages("devtools")

devtools::install_github("Kohze/fireData")
library(fireData)
download(projectURL = "https://firedata-b0e54.firebaseio.com/", fileName = "main/-KxwWNTVdpIXFRZwGMkH")
```

● Medium: Setup your own firebase cloud, create your own user account via firedata and upload/download a sample dataset.

```
createUser(projectAPI = "AIzaSyA6fjtX9P6y-4GsTq_P92uIfkn4LjXBI7E", email = "gnehsaijuhz@hotmail.com",
password = "gsoc2018")
```

```
auth(projectAPI = "AIzaSyA6fjtX9P6y-4GsTq_P92uIfkn4LjXBI7E", email = "gnehsaijuhz@hotmail.com", password = "
gsoc2018")

sampleData <- read.csv(file='xgb_valid.csv',sep=',',header=T)
upload(x = sampleData, projectURL = "https://gsoc2018-d05d8.firebaseio.com/", directory = "sample")
```

● Hard: Add examples of the package use to the Rd files in the following format

```
\name{upload}
\alias{upload}
%- Also NEED an '\alias' for EACH other topic documented here. \title{
Upload a dataset to firebase
}
\description{

Upload a dataset to the specified directory in firebase
}
\usage{
upload(x, projectURL, directory = "main", token = "none") }

\arguments{ \item{x}{

X should point to the dataset that is to be uploaded }

\item{projectURL}{
projectURL is the Firebase URL

} \item{directory}{

Directory to put x inside the firebase project }

\item{token}{ Firebase token

} }

\details{ } \value{

\item{result}{the combined string of the directory and the http status} }

\references{
}
\author{ Jiasheng Zhu }

\note{
Please auth first or provide a token
}
\examples{
function (x, projectURL, directory = "main", token = "none") {

output = fileConversion(x) if (token == "none") {

Response = httr::POST(paste0(projectURL, "/", directory, ".json"), body = jsonlite::toJSON(output, auto_unbox =
TRUE))

} else {
```

```
Response = httr::POST(paste0(projectURL, "/", directory, ".json?auth=", token), body = jsonlite::toJSON(output,
auto_unbox = TRUE))

}

return(paste0(directory, "/", httr::content(Response)$name)) }

<environment: namespace:fireData> }
```

Finishing these tests is a demonstration of the minimum skills needed to start the Firedata projects, including basic R programing, Firebase uses and Rd uses.

A Github link for all the files is here: https://github.com/19hourz/gsoc2018.