Birla Institute of Technology and Science**,** Pilani



**Cross Platform Application Development**

**Assignment -1**

By Group#5

TANYA AGGARWAL 2021MT93286

ONKAR C DESHPANDE 2021MT93590

MARWAL MOHIT TARACHAND 2021MT93541

**Submitted to**

**Pravin Yashwant Pawar**

# Objective:

To create working prototype of cross-platform application (involving both frontend and backend components with interactions between them) for the specified user stories using the technologies / frameworks discussed in the classroom sessions

# **Problem STATEMENT:**

Design and develop a mobile & web-based vaccine tracking system for administration, tracking and management of vaccination for schools.

Following use cases needs to be addressed –

1. Student Management – Adding & managing student data for vaccination. There should be provision for bulk upload.
2. Vaccine Drive Management – Conducting vaccination drives , and managing these drives . Vaccination status for Students who are part of drive should also be updated.
3. Reports – Generating report based on student and vaccination data based on various filters.
4. Dashboard – Summary of school`s vaccination status.

# **Assumptions:**

Following assumptions are taken while developing this application –

* User management is out of scope of this solution , and hence it is assumed that user is logged in and has sufficient privileges to use various functionalities.
* School administers two vaccines – Covaxin & Covishield. This assumption is for avoiding any need of vaccine master data and associated tooling. Note – There should not be hardcoding of vaccine in any logic.
* There is no approval workflow implemented for vaccination drive. It is assumed that a drive can be pre-approved , or a user can change status of drive to approved.
* Completed or Cancelled are terminal state of drive , and there cannot be any change in drive at this state.
* Students could be vaccinated from outside of school drive also.

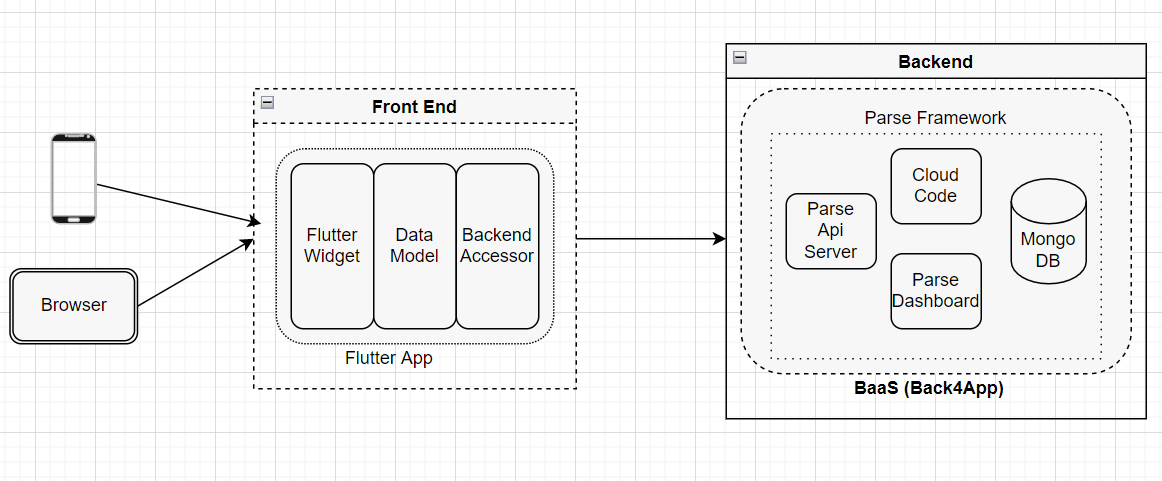
# **Solution Overview:**

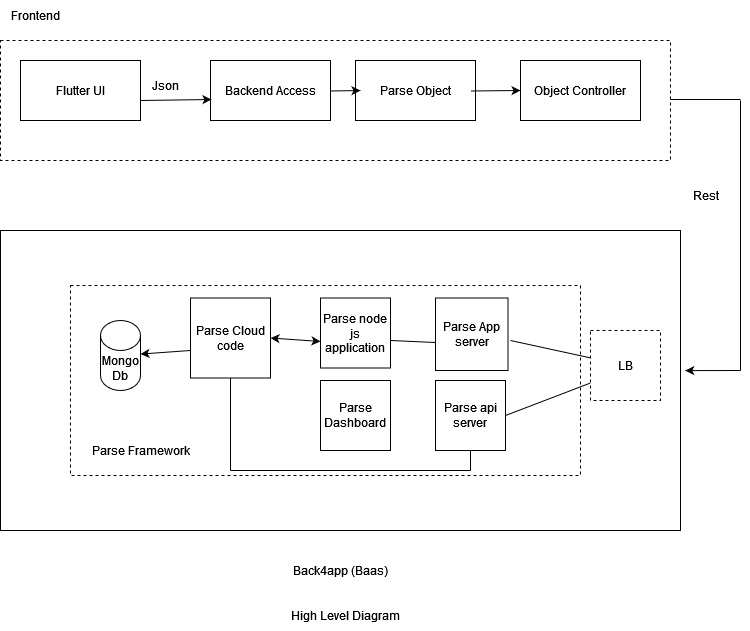
This solution uses cross platform application framework (Flutter) and back end as a service (Baas) provided by back4app.

**High Level Architecture –**

A mobile or web user interacts with frontend built using **Flutter** , while frontend interacts with backend which is provided by a backend service provider [**back4app**](https://blog.back4app.com/backend-as-a-service-baas/).

Back4app can be considered as a managed **Parse open source framework** and sufficient tooling on top of it.



**A deeper dive into high level architecture – **

# Tech Stack:

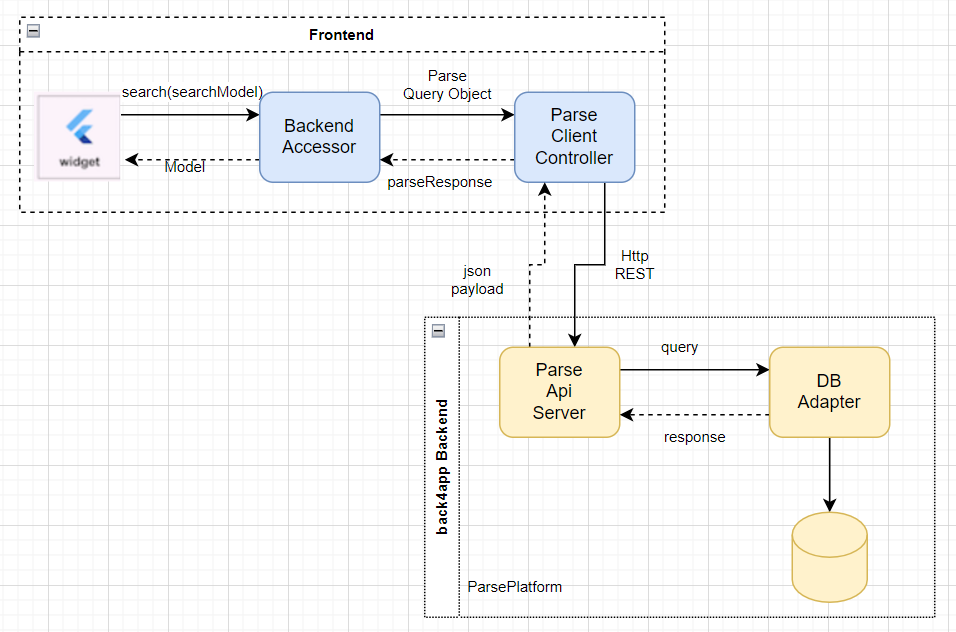
|  |  |
| --- | --- |
| Frontend | Flutter Cross Platform application framework |
| Parse SDK for Flutter (<https://github.com/parse-community/Parse-SDK-Flutter>) |
| Android Studio |
| Android Virtual Device (AVD) |
| Chrome Browser |
|  |  |
| Backend | back4app BaaS Parse Platform (<https://parseplatform.org/>) |
| Database - back4app Real Time Database Service |
| Business Logic - back4app Cloud functions |
| File Storage - back4app FileStorage service |
| back4app JS console |

# Software working in Detail

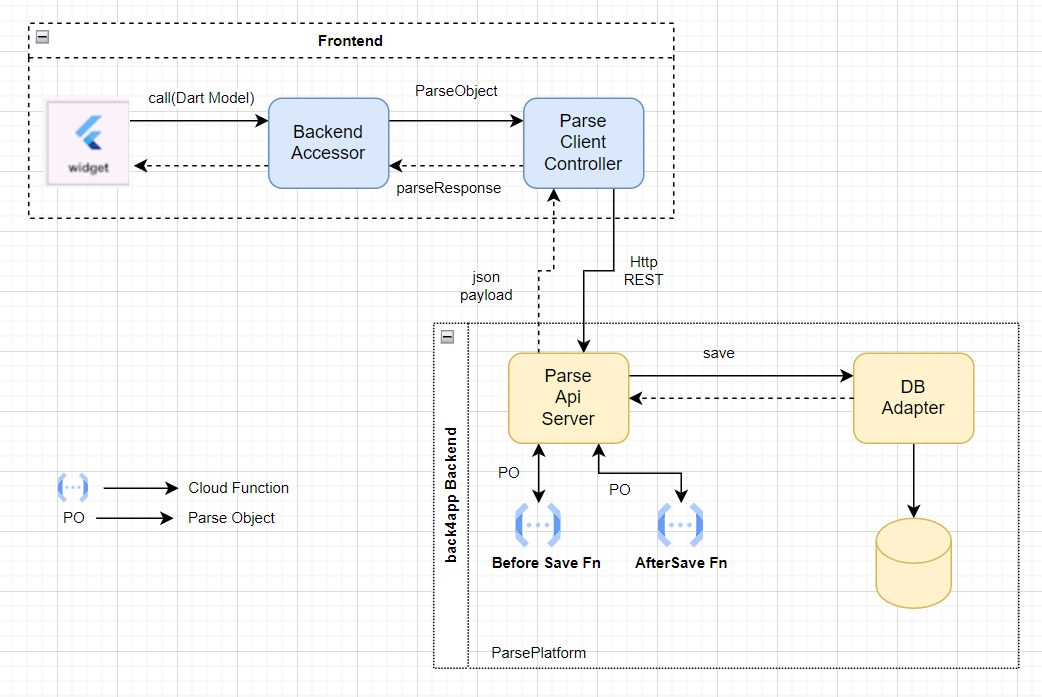
We have created a pattern for all the flows for maximizing reusability and decoupling. All system flows could be divided into 2 categories –

* Query – that is getting the data and displaying.
* Command – mutating operations like save, edit , bulkupload.

Pattern For Query



Pattern For Command (Mutating Operations)

****

**Reason For Using BackendAccessor** – backend accessor decouples Flutter UI layer from Parse , an internal model is passed from flutter UI layer to backend accessor which in turn uses Parse sdk to invoke correct ParseObject. With this decoupling , backend accessor can switch to any other backend (like pure http REST).

**Reason for Using client side flutter-Parse sdk –** Parse Object abstract asynchronous REST communication over network. If we would have not used this sdk , boiler plate code for handling asynchronous REST communication & json data marshalling-unmarshalling would have fallen onto us.

**Reference taken for understanding Parse Client Side working –**

<https://engineering.fb.com/2015/08/13/core-data/the-parse-sdk-what-s-inside/>

So on the wire its REST over Http. REST api spec can be found at –

<https://github.com/OnkarDeshpande-BITS/CPADAssignment-1/blob/main/documentation/swagger.json>

**Manage Student Flow**

Data Model – <https://github.com/OnkarDeshpande-BITS/CPADAssignment-1/tree/main/documentation/DatabaseModel.xlsx>

|  |  |  |
| --- | --- | --- |
| **StudentVaccinationDetails** | | |
| **Attributes** | **Description** | **SampleData** |
| objectId | Unique Identifier of a row in DB |  |
| studentId | Student Identifier like roll no ,  which identifies student in other system as well | 2018-C1-10054 |
| name | Name of student |  |
| aadharNo | Govt UUID of student. Needed to update vaccination details with govt |  |
| dob | Date of birth of student |  |
| isVaccinated | Vaccinated or not |  |
| doseDetails | Its Json array of administered  vaccines | [{"doseNo":1,"date":1122020,  "name":"Covaxine","batchNo":"111222"} ,{"doseNo":2,"date":1122021,  "name":"Covaxine","batchNo":"333222"}] |
| noOfDoses | for reporting , derived  from 'doseDetails' |  |
| primaryVaccineName | derived field from 'doseDetails'  for validating subsequent vaccine doses |  |
| lastVaccineDt | derived field from 'doseDetails'  for reporting purpose and enforcing date based validation |  |
| createdAt | Audit field |  |
| updatedAt | Audit field |  |
| schoolDriveId | UUID of vaccination drive conducted  in school. |  |

1. Create Student Call flow

Request Response Details –

Request & Response are summarized below –

CallFlow –

**Request**

URL

https://parseapi.back4app.com/classes/StudentVaccinationDetails

Method

POST

Headers

X-Parse-Application-Id: XXXXX

X-Parse-REST-API-Key: XXXX

Content-Type: application/json

Body

A JSON document with the key-value pairs that represent your object's data according to the supported fields.

Example -

{ \"studentId\":\"A string\",\"name\":\"A string\",\"aadharNo\":\"A string\",\"isVaccinated\":true,\"noOfDoses\":1,\"primaryVaccineName\":\"A string\",\"schoolDriveId\":\"A string\",\"dob\":1,\"doseDetails\":[ 1, \"a string\" ],\"lastVaccineDt\":1 }

**Success Response**

**Status**

**201 Created**

Headers

Location: https://parseapi.back4app.com/classes/StudentVaccinationDetails/MyNewObjectId

The Location header will contain the endpoint of the newly-created object.

Body

A JSON document with the objectId and createdAt fields of the newly-created object.

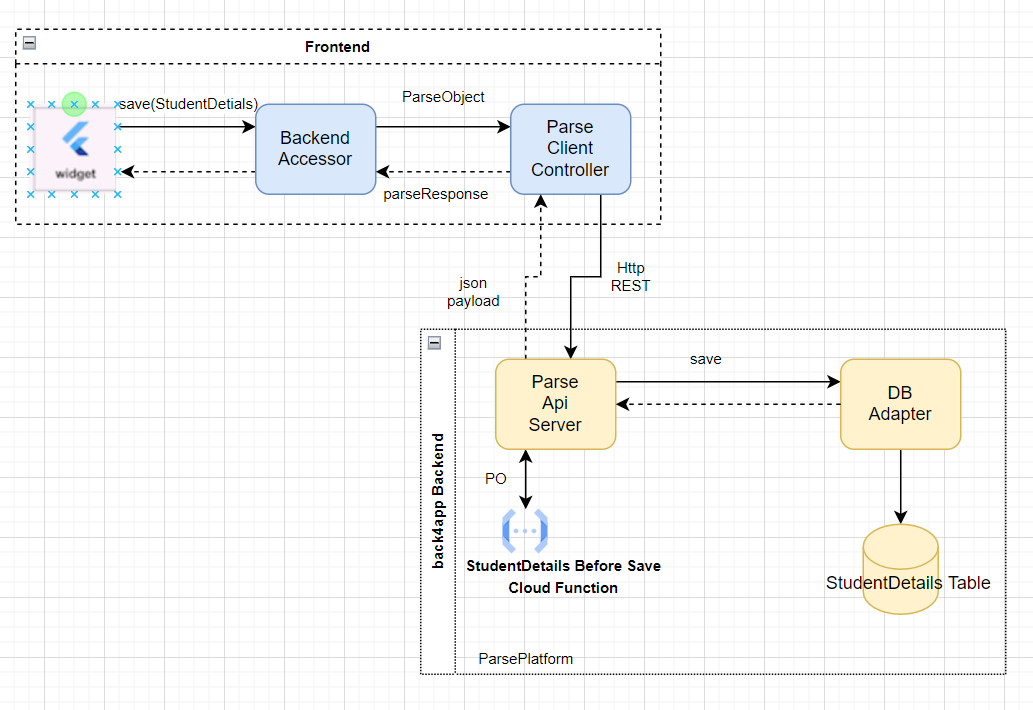
Example -

{

"objectId": "4BwpMWdCnm",

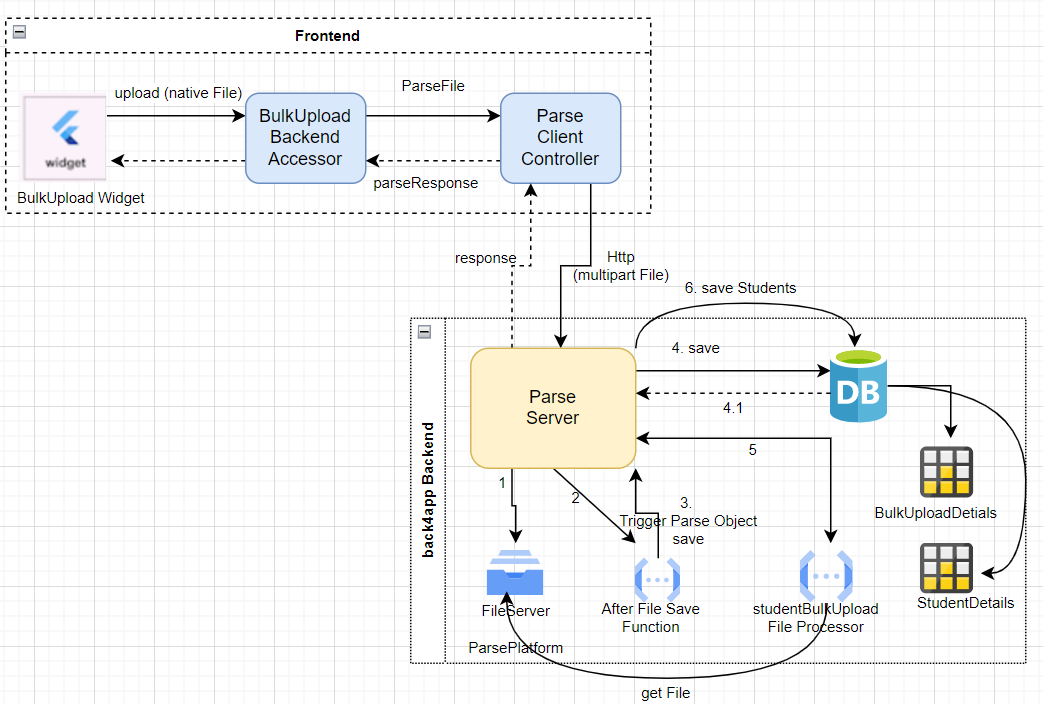
"createdAt": "2018-11-06T00:52:01.520Z"

}



*\*\* call flow is similar to ‘command’ patter stated above.*

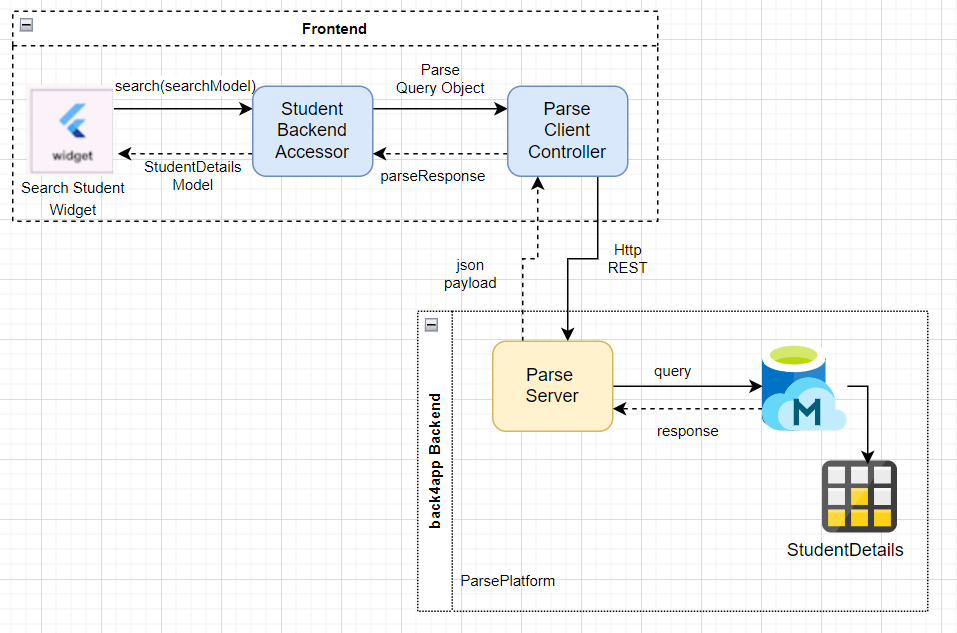
Bulk Upload Student Flow –



*Explanation of server side flow ( follow numbers marked on arrows)*

1. File is saved on file server and acknowledgement is sent back to parse server.
2. Parse Server calls , configured ***after FileSave Cloud Code Function*** . This cloud function save file meta data like name & url with other meta info in BulkUploadDetails table via ParseObject api. This is done so file could be referred later for retrieval or other operation.
3. As mentioned Cloud function triggers Parse Object save on Parse Server.
4. BulkUpload Details saved to real time db table.
5. After save cloud function is invoked by Parse Server after successful save of BulkUpload Details object. This cloud function has logic to read the file from file server and create StudentDetails via Parse Server.
6. Save StudentDetails.

Query StudentDetails

****

***Its inspired by query flow already described above.***

**Request**

URL

https://parseapi.back4app.com/classes/StudentVaccinationDetails

Method

GET

Headers

X-Parse-Application-Id: XXXX

X-Parse-REST-API-Key: XXXX

Parameters

A where URL parameter constraining the value for keys. It should be encoded JSON.

Example -

where={ \"studentId\":\"A string\",\"name\":\"A string\", \"isVaccinated\":true,\"schoolDriveId\":\"A string\"}"

**Success Response**

**Status**

**200 OK**

Headers

content-type: application/json;

Body

a JSON object that contains a results field with a JSON array that lists the objects.

Example -

{

"results": [

{

"objectId": "zJxVP17mTi",

"createdAt": "2018-10-31T14:16:13.616Z",

"updatedAt": "2018-11-07T12:12:20.758Z",

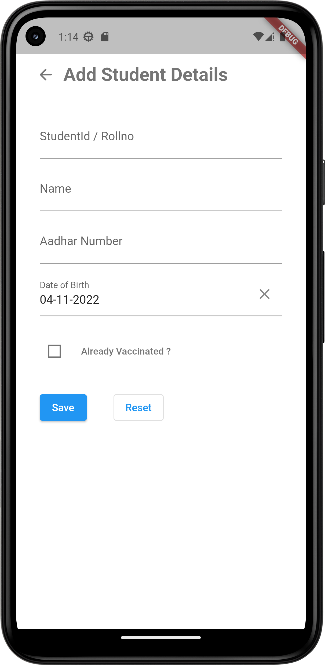
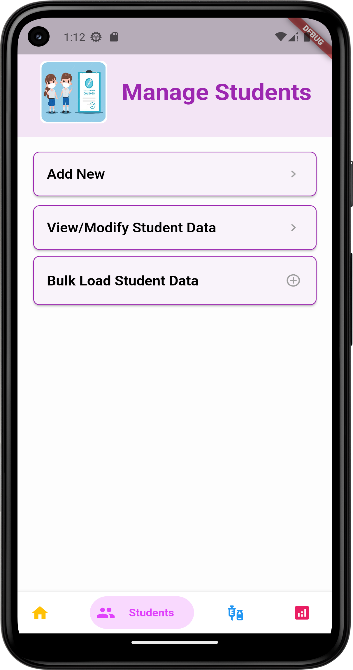
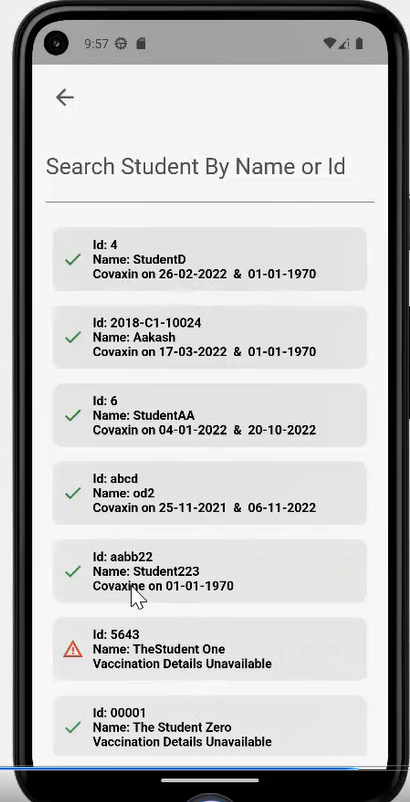
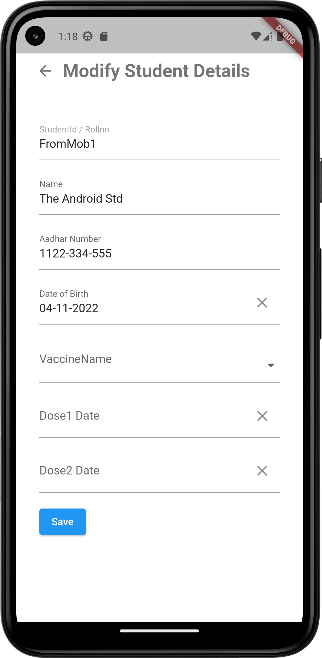
"studentId": \"A string\","name": \"A string\","aadharNo": \"A string\","isVaccinated": true,"noOfDoses": 1,"primaryVaccineName": \"A string\","schoolDriveId": \"A string\","dob": 1,"doseDetails": [ 1, \"a string\" ],"lastVaccineDt": 1

}

]

}

**Screen shots for Manage Student from Demo –**

****

**Manage Vaccination Drive**

Data Model – <https://github.com/OnkarDeshpande-BITS/CPADAssignment-1/tree/main/documentation/DatabaseModel.xlsx>

|  |  |  |
| --- | --- | --- |
| **VaccinationDrive** | | |
| **Attributes** | **Description** | **SampleData** |
| objectId | Unique Identifier of a row in DB |  |
| driveName | Meaningful name of drive to identify it by name | 8Oct\_Rainbow\_covishield\_300 |
| driveDt | Date on which drive is/was conducted |  |
| state | state in which drive is - draft ,  pending , complete , cancelled | DRAFT - drive created in system but not approved PENDING - approved drive , pending execution COMPLETE - completed CANCELLED - drive was cancelled |
| totalDoses | total doses in particular drive |  |
| vaccineDetails | details of vaccines in drive | [{"name":"Covaxine","doses":300},  {"name":"Covishield","doses":600}] |

Request Response Details –

Create VaccinationDrive

*\*\* Please note call flow for create follows ‘command pattern’ hence not putting similar diagram.*

**Request**

URL

https://parseapi.back4app.com/classes/VaccinationDrive

Method

POST

Headers

Content-Type: application/json

Body

A JSON document with the key-value pairs that represent your object's data according to the supported fields.

Example -

{ \"driveDt\":1,\"state\":\"A string\",\"totalDoses\":1,\"driveName\":\"A string\",\"vaccineDetails\":[ 1, \"a string\" ] }

**Success Response**

**Status**

**201 Created**

Headers

Location: https://parseapi.back4app.com/classes/VaccinationDrive/MyNewObjectId

Body

Example -

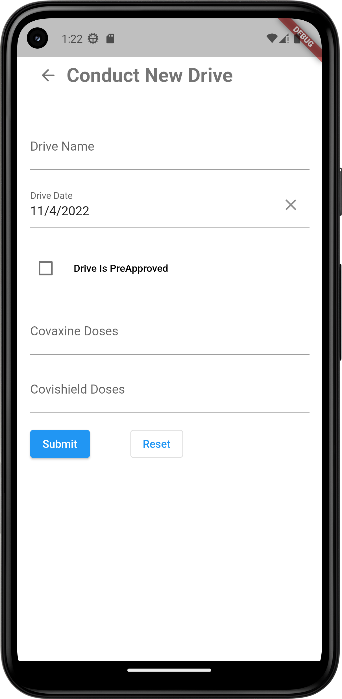
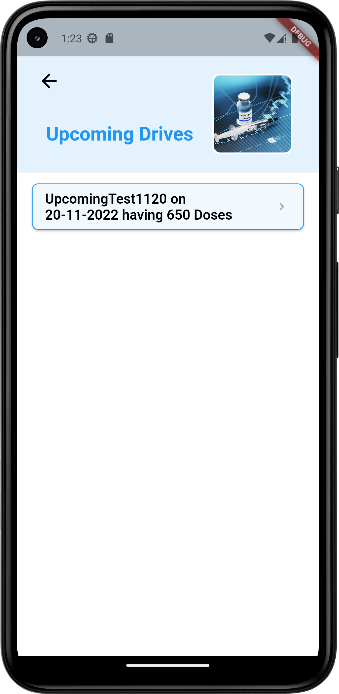
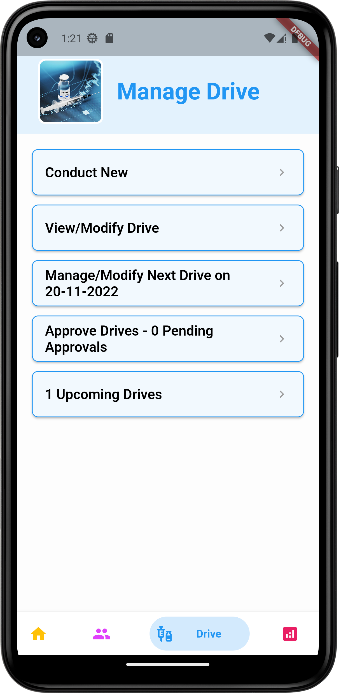
{

"objectId": "4BwpMWdCnm",

"createdAt": "2018-11-06T00:52:01.520Z"

}

Snapshots from demo –



Query Vaccination Drive

**Request**

URL

https://parseapi.back4app.com/classes/VaccinationDrive

Method

GET

Headers

X-Parse-Application-Id: XXXXX

X-Parse-REST-API-Key: XXXXX

Parameters

A where URL parameter constraining the value for keys. It should be encoded JSON.

Example -

where={ \"driveDt\":1,\"state\":\"A string\",\"driveName\":\"A string\" }

**Success Response**

**Status**

**200 OK**

Headers

content-type: application/json;

Body

a JSON object that contains a results field with a JSON array that lists the objects.

{

"results": [

{

"objectId": "zJxVP17mTi",

"createdAt": "2018-10-31T14:16:13.616Z",

"updatedAt": "2018-11-07T12:12:20.758Z",

"driveDt": 1,"state": \"A string\","totalDoses": 1,"driveName": \"A string\","vaccineDetails": [ 1, \"a string\" ]

}

]

}

**Dashboard & Generate Reports**

Dashboard is prepared for showing summarized view to user. It collects data from different tables and shows in a meaningful way to make more sense.

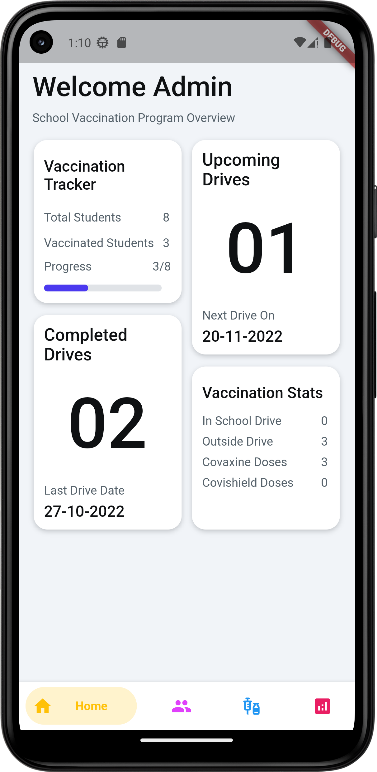
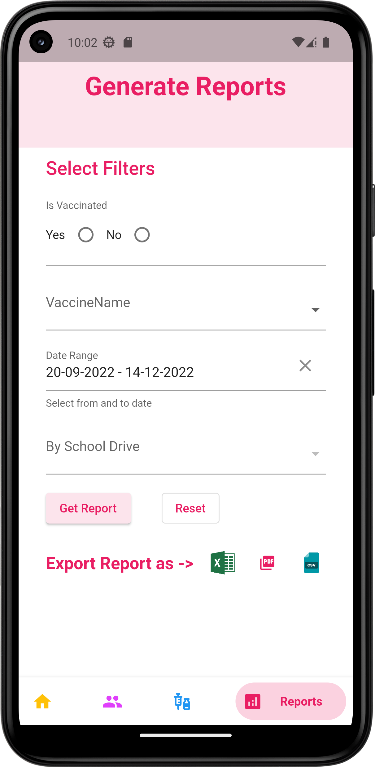
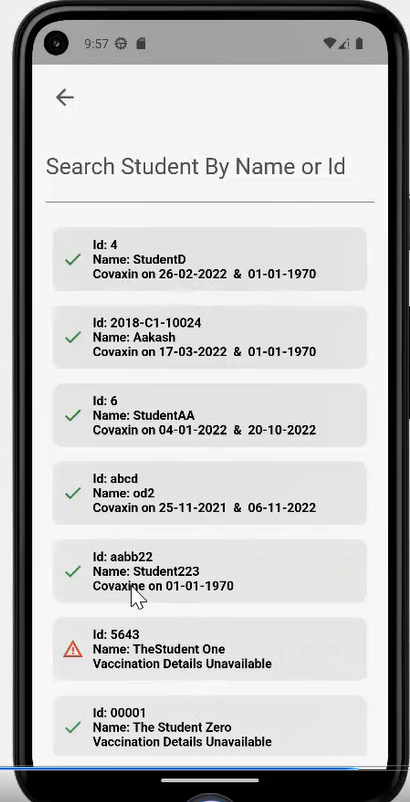
Dashboard also follows ‘query pattern’ as described above , with only difference , that it simultaneously queries multiple sources.

Generate Reports provides ability to use certain search criteria to find relevant data. Various reports could be extracted using search criteria provided. Few Reports could be –

* Report of all non vaccinated students
* Report for students vaccinated within data range.
* Report by vaccine name
* Report of students vaccinated in particular drive conducted by school.

Information displayed in report consist of – studentId , studentName , Vaccination status, Vaccine administered and on which dates. There is a provision given to export report to excel . pdf or csv.

Screen shots from demo –

 ****

# Git Repo And Video Link:

Git - <https://github.com/OnkarDeshpande-BITS/CPADAssignment-1>

API Documentation - <https://github.com/OnkarDeshpande-BITS/CPADAssignment-1/blob/main/documentation/swagger.json>

Demo Video Link - <https://drive.google.com/drive/folders/1dupA4PUnOH1TYzxxBjFh_vQPIYLcSSVq?usp=share_link>

Screen Shots Link –

<https://github.com/OnkarDeshpande-BITS/CPADAssignment-1/tree/main/documentation/Snapshots>

Shortcut for frontend code - <https://github.com/OnkarDeshpande-BITS/CPADAssignment-1/tree/main/frontend/vaccination_mgmt>

Shortcut for backend code - <https://github.com/OnkarDeshpande-BITS/CPADAssignment-1/tree/main/backend>

Data Model - https://github.com/OnkarDeshpande-BITS/CPADAssignment-1/blob/main/documentation/DatabaseModel.xlsx