DEVELOPMENET PHASE REPORT

API/BACKEND & MOBILE APP

Unlimited Cloud Pte Ltd

SUMMARY

Development phase of the solution that includes creating API, backend, web portal and mobile application has been concluded. With the feedback and constant discussion, the solution has been shaped up to meet all the requirements. The UI has also been changing and is now is a stable final state that meets the ease of use for users. Various features have been added and a detail explanation of all the features included in the solution has been described below. Since simultaneous work had been done by API/ Web portal and app development team, all the features have completed in both app and web side at the same time. To make the system stable and bug free a series of testing will be carried out.

ACTION(S) TAKEN

API CREATED

API stands for Application Programming Interface. An API is a software intermediary that allows two applications to talk to each other. In other words, an API is the messenger that delivers the request to the provider that user is requesting it from and then delivers the response back to them.

One of the chief advantages of APIs is that they allow the abstraction of functionality between one system and another. An API endpoint decouples the consuming application from the infrastructure that provides a service. As long as the specification for what the service provider is delivering to the endpoint remains unchanged, the alterations to the infrastructure behind the endpoint should not be noticed by the applications that rely on that API.

BACKEND & WEB PORTAL DESIGNED

Along with creation of APIs, backend of the solution which is based on PHP Laravel is also developed and designed according the requirements. Views for different modules have been created and necessary actions were taken so that details could be displayed in proper order. Detail description of the actions done for each module is listed below:

MOBILE APPLICATION CREATED

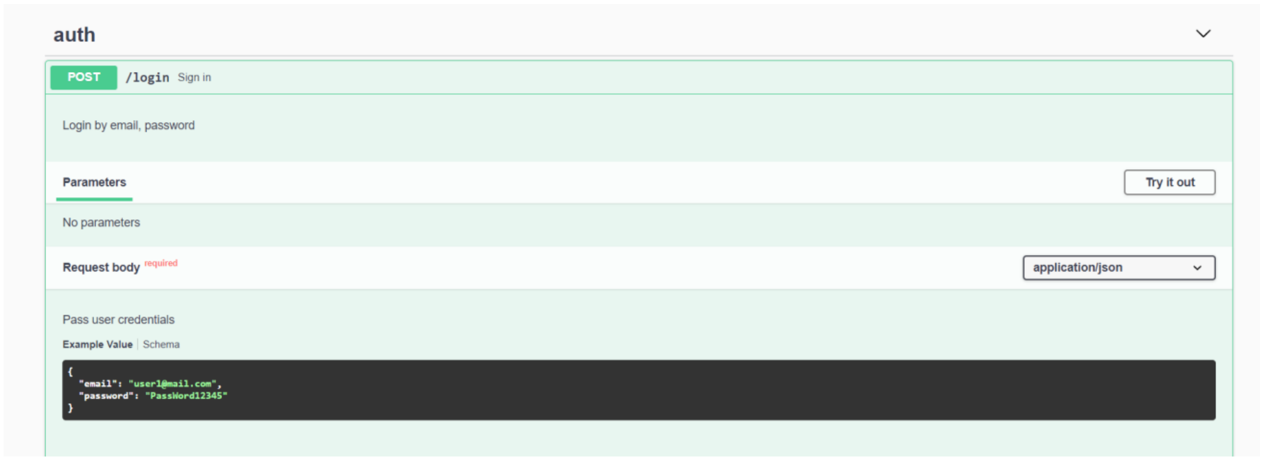
Another team of flutter developers had been working simultaneously to create an android application for our cash for work solution. This application would mostly be used in field by facilitators to collect various data. Clean and understandable designing had also been made so that it gets clear for anyone to use the application. While backend and APIs were created, this team would simultaneously be adding up features in the mobile application so that we meet the project timeline. Detail description of the action taken for each module is described below:

LOGIN MODULE

API FOR LOGIN

To handle the requests on login, a login API that manages all the login requests is created. Details of the API are:

End Point: (baseurl) /login
Method: POST
Request Body: Email, Password



/login Sign in

Login by email, password

Parameters

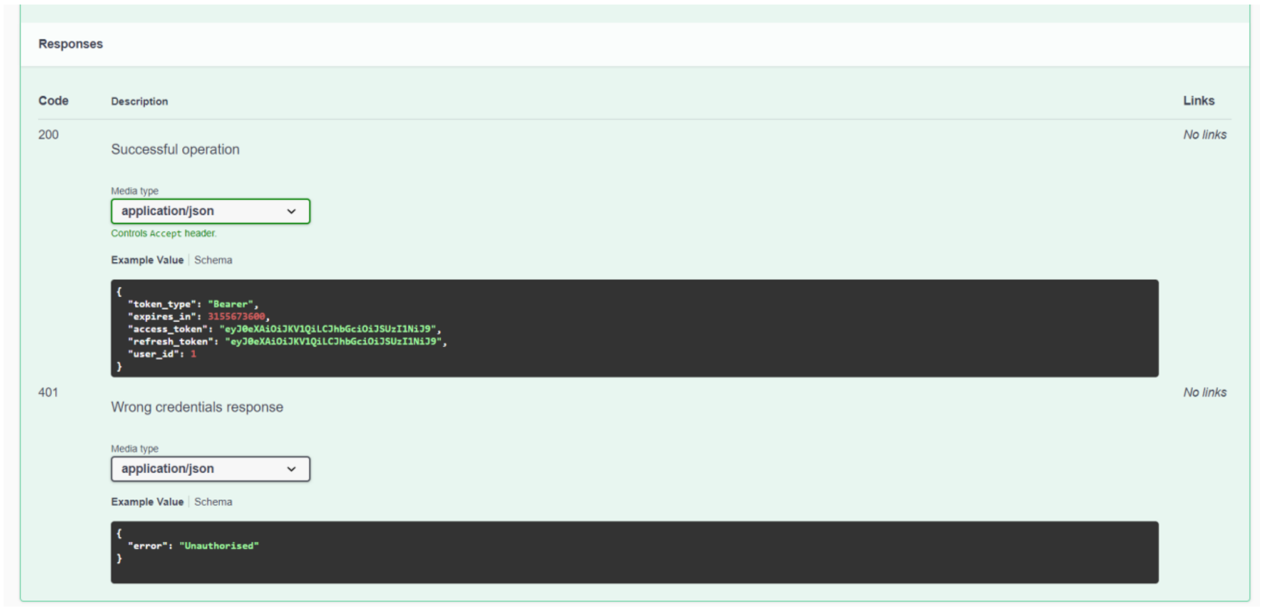
No parameters

Request body required — application/json

Pass user credentials

Example Value | Schema

```
{
  "email": "user1@mail.com",
  "password": "PassWord12345"
}
```



Responses

Code: 200 — Successful operation — No links

Media type: application/json
Controls Accept header.

Example Value | Schema

```
{
  "token_type": "Bearer",
  "expires_in": 3155673600,
  "access_token": "eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9",
  "refresh_token": "eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9",
  "user_id": 1
}
```

Code: 401 — Wrong credentials response — No links

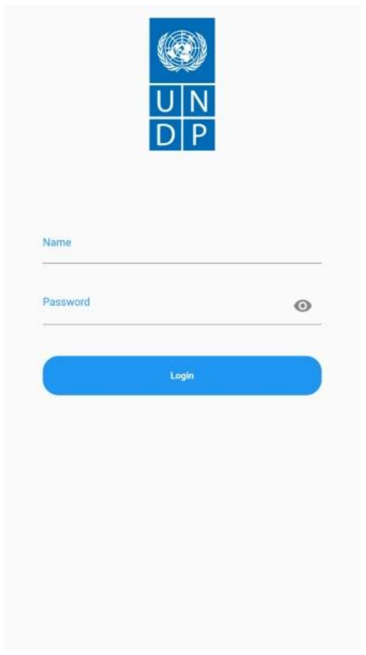
Media type: application/json

Example Value | Schema

```
{
  "error": "Unauthorised"
}
```

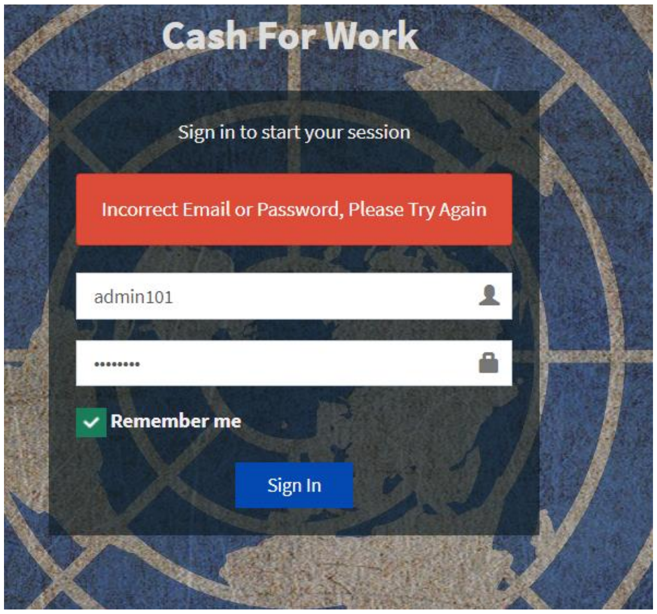
VIEWS FOR LOGIN

To be able to login for the users a login page has also been created on both platforms. Two fields to enter username and password is given when users will have to enter the login credentials assigned to them. If correct credentials are passed then user will be directed to next screen where they will be able to perform other actions but if not, then error will be displayed to them. The screenshot below demonstrates the login view.



In the screenshot shown alongside, we can see that incorrect credentials were passed in the username and password fields. Hence, error was displayed and the user was not allowed to proceed to other screen . As long as correct details are passed user will be shown the error and will remain on this page.

Fig. Login for Mobile App is also shown in the image here.


Cash For Work

Sign in to start your session

Incorrect Email or Password, Please Try Again

admin101

••••••••

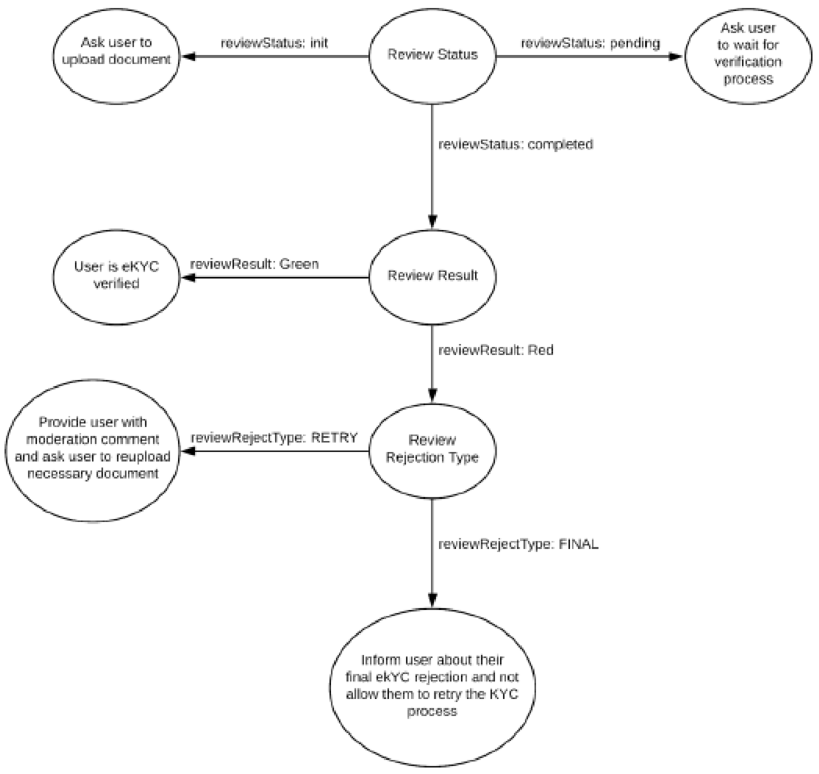
✓ Remember me

Similar to the login page on the web portal, users (facilitator) will have to provide correct credential to be able to view other features and proceed to next screen.

WORKER LIST & REGISTRATION

WORKER REGISTRATION PROCESS

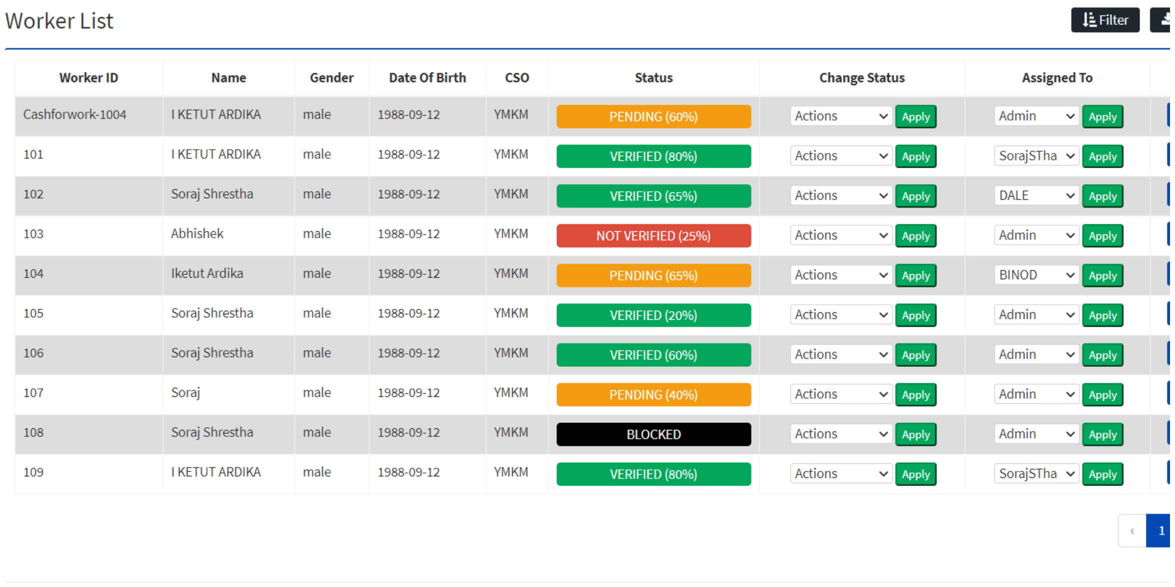
For the worker registration process, team has used machine learning techniques and cognitive service to OCR data from national ID card and verify user via their image on document and a selfie. A match of the OCR data and manual data and match of their document image and selfie image will be made for the verification process. Flowchart below represents the flow that the user have to make to be registered as a verified user.


Ask user to upload document
reviewStatus: init
Review Status
reviewStatus: pending
Ask user to wait for verification process
reviewStatus: completed
User is eKYC verified
reviewResult: Green
Review Result
reviewResult: Red
Provide user with moderation comment and ask user to reupload necessary document
reviewRejectType: RETRY
Review Rejection Type
reviewRejectType: FINAL
Inform user about their final eKYC rejection and not allow them to retry the KYC process

Detail action made on the entire worker registration flow is described below:

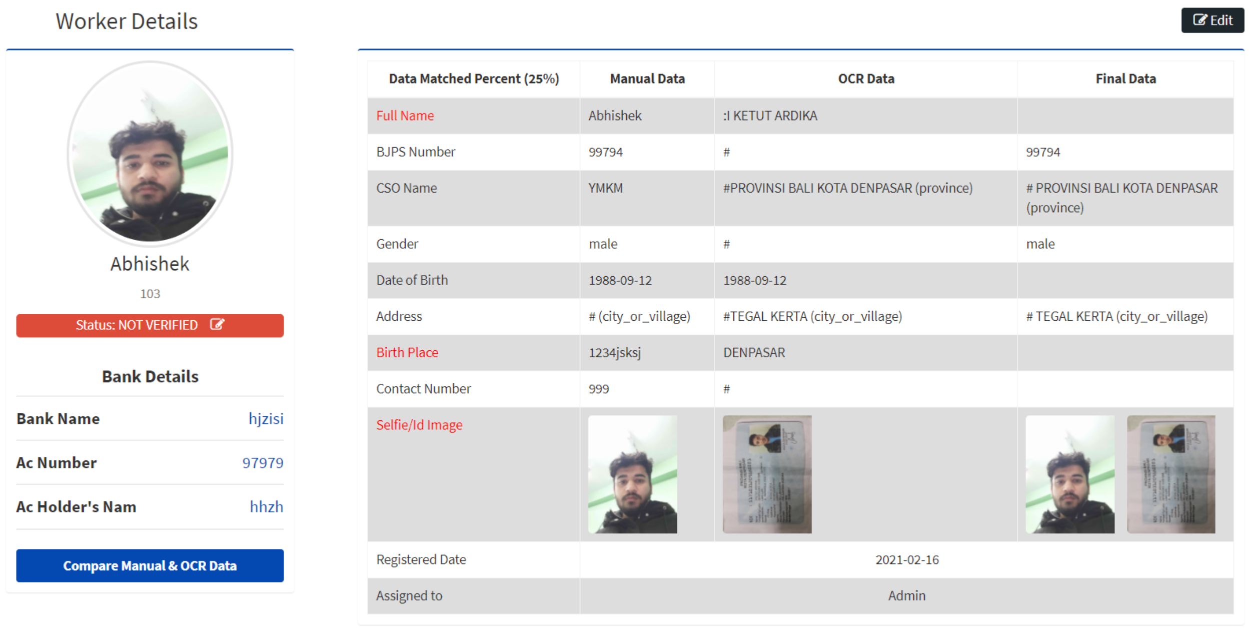
- Upload document Id and selfie picture.
- Wait until the verification process completes.
- A complete match will reflect him as verified worker automatically. No user(admin) action will be required for such workers.

- If partial data is matched then such workers will be categorized as Pending Workers. A red mark on the items that are not matched is displayed on the dashboard. Dashboard User (Admins) will now have to make necessary changes on the fields that are red.
- If there is no match in data then such worker will be listed as Unverified user. The admin will now have to decide whether to register the worker entirely from beginning or upload all the docs again.
- There is also a Blocked Type worker category where admins can change any worker type to this category. Such worker will not be allowed to perform any action when they are categorized in this type.



Worker ID	Name	Gender	Date Of Birth	CSO	Status	Change Status	Assigned To
Cashforwork-1004	I KETUT ARDIKA	male	1988-09-12	YMKM	PENDING (60%)	Actions Apply	Admin Apply
101	I KETUT ARDIKA	male	1988-09-12	YMKM	VERIFIED (80%)	Actions Apply	SorajSTha Apply
102	Soraj Shrestha	male	1988-09-12	YMKM	VERIFIED (65%)	Actions Apply	DALE Apply
103	Abhishek	male	1988-09-12	YMKM	NOT VERIFIED (25%)	Actions Apply	Admin Apply
104	Iketut Ardika	male	1988-09-12	YMKM	PENDING (65%)	Actions Apply	BINOD Apply
105	Soraj Shrestha	male	1988-09-12	YMKM	VERIFIED (20%)	Actions Apply	Admin Apply
106	Soraj Shrestha	male	1988-09-12	YMKM	VERIFIED (60%)	Actions Apply	Admin Apply
107	Soraj	male	1988-09-12	YMKM	PENDING (40%)	Actions Apply	Admin Apply
108	Soraj Shrestha	male	1988-09-12	YMKM	BLOCKED	Actions Apply	Admin Apply
109	I KETUT ARDIKA	male	1988-09-12	YMKM	VERIFIED (80%)	Actions Apply	SorajSTha Apply

Img. Worker List in different Categories



Worker Details

Abhishek

103

Status: NOT VERIFIED

Bank Details

Bank Name	hjzisi
Ac Number	97979
Ac Holder's Nam	hhzh

Compare Manual & OCR Data

Edit

Data Matched Percent (25%)	Manual Data	OCR Data	Final Data
Full Name	Abhishek	:I KETUT ARDIKA	
BJPS Number	99794	#	99794
CSO Name	YMKM	#PROVINSI BALI KOTA DENPASAR (province)	# PROVINSI BALI KOTA DENPASAR (province)
Gender	male	#	male
Date of Birth	1988-09-12	1988-09-12	
Address	# (city_or_village)	#TEGAL KERTA (city_or_village)	# TEGAL KERTA (city_or_village)
Birth Place	1234jsksj	DENPASAR	
Contact Number	999	#	
Selfie/Id Image			
Registered Date	2021-02-16		
Assigned to	Admin		

Img. Worker Detail of Unverified Users with Unmatched fields denoted in red.

A detail procedure of machine learning and the training module worked so far has been described below:

- The use of the cognitive service (Image predictor) will filter the correctness of the document. Only verified National Id can be taken to register worker. Any other document type is not understood by the system hence this is the first step to pass to register the worker.
- Cognitive service (form recognizer labelling tool) is used to train the Indonesian primary Identity card . Currently by collecting those cards from internet so that we have created the model which helps to create train model with the help of docker.
- By accepting the Identity Document from the front end the application then process that image against the created train model which returns the user details . These details are now stored database. All images are stored on azure blob storage.

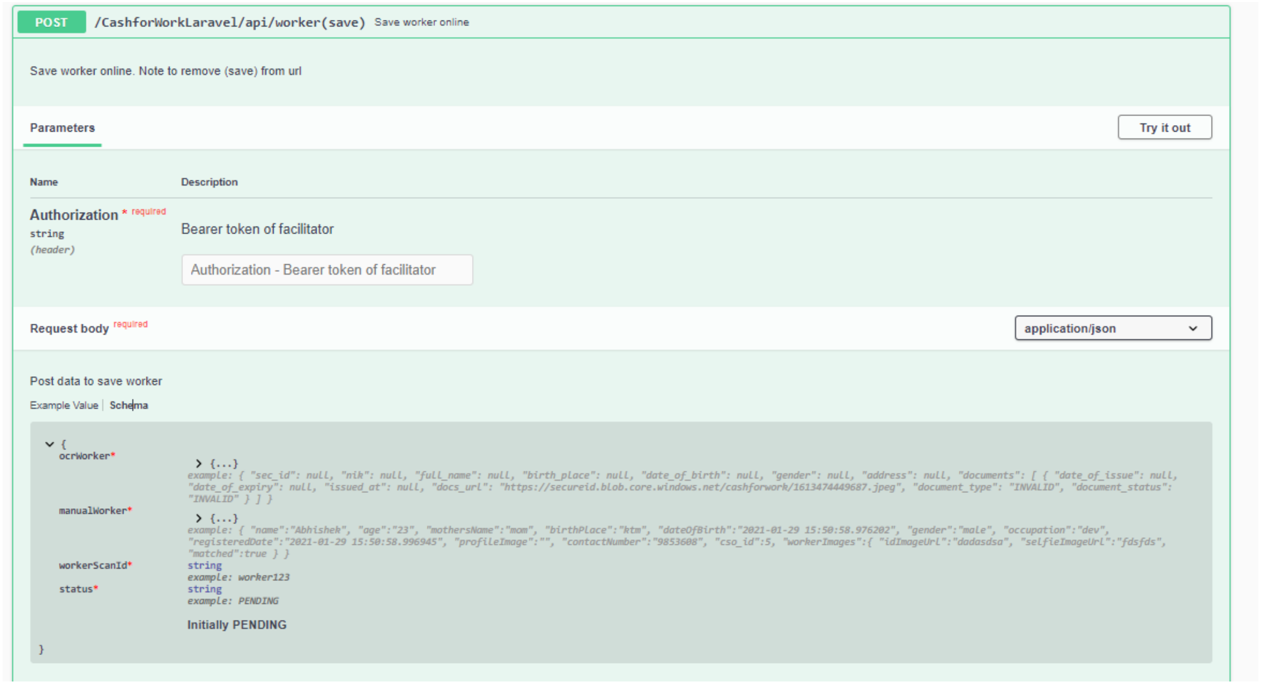
Form Recognizer (labelling Tool):

Form Recognizer is a cognitive service that lets you build automated data processing software using machine learning technology. Identify and extract text, key/value pairs, selection marks, tables, and structure from your documents—the service outputs structured data that includes the relationships in the original file, bounding boxes, confidence and more. When we train with labelled data, the model does supervised learning to extract values of interest, using the labelled forms you provide. This results in better-performing models and can produce models that work with complex forms or forms containing values without keys.

API FOR WORKER REGISTRATION

To train and extract user details from identity document:
Request method:POST
(baseurl)/api/cfw/extract/{id-document}
Id-document= NID for Indonesian Identity card

API FOR REGISTERING WORKER ONLINE



```
POST /CashforWorkLaravel/api/worker(save) Save worker online

Save worker online. Note to remove (save) from url

Parameters                                                    Try it out

Name                 Description
Authorization * required   Bearer token of facilitator
string
(header)             Authorization - Bearer token of facilitator

Request body required                                         application/json

Post data to save worker
Example Value | Schema

{
  ocrWorker*        > {...}
                    example: { "sec_id": null, "nik": null, "full_name": null, "birth_place": null, "date_of_birth": null, "gender": null, "address": null, "documents": [ { "date_of_issue": null, "date_of_expiry": null, "issued_at": null, "docs_url": "https://secureid.blob.core.windows.net/cashforwork/1613474449687.jpeg", "document_type": "INVALID", "document_status": "INVALID" } ] }
  manualWorker*     > {...}
                    example: { "name":"Abhishek", "age":"23", "mothersName":"mom", "birthPlace":"ktm", "dateOfBirth":"2021-01-29 15:50:58.976202", "gender":"male", "occupation":"dev", "registeredDate":"2021-01-29 15:50:58.996945", "profileImage":"", "contactNumber":"9853608", "cso_id":5, "workerImages":{ "idImageUrl":"dadasdsa", "selfieImageUrl":"fdsfds", "matched":true } }
  workerScanId*     string
                    example: worker123
  status*           string
                    example: PENDING

                    Initially PENDING
}
```

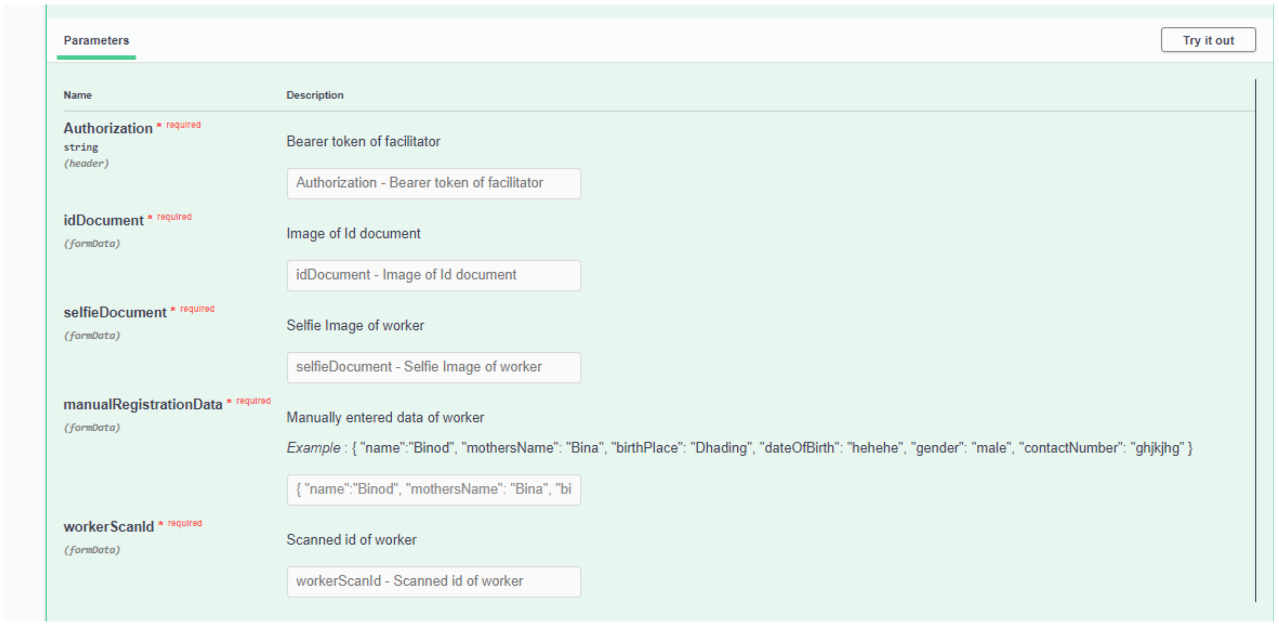
As shown above the URL of the API to register workers during online mode of the application is
URL: (baseURL) /CashforWorkLaravel/api/worker
Method: POST

The request body needs data from server that does the OCR and also data from manual registration. Hence an example of request body is shown below:

```
{
  "ocrWorker": {
    "sec_id": null,
    "nik": null,
    "full_name": null,
    "birth_place": null,
    "date_of_birth": null,
    "gender": null,
    "address": null,
    "documents": [
      {
        "date_of_issue": null,
        "date_of_expiry": null,
        "issued_at": null,
        "docs_url": "https://secureid.blob.core.windows.net/cashforwork/1613474449687.jpeg",
        "document_type": "INVALID",
        "document_status": "INVALID"
      }
    ]
  },
  "manualWorker": {
    "name": "Abhishek",
    "age": "23",
    "mothersName": "mom",
    "birthPlace": "ktm",
    "dateOfBirth": "2021-01-29 15:50:58.976202",
    "gender": "male",
    "occupation": "dev",
    "registeredDate": "2021-01-29 15:50:58.996945",
    "profileImage": "",
    "contactNumber": "9853608",
    "cso_id": 5,
    "workerImages": {
      "idImageUrl": "dadasdsa",
      "selfieImageUrl": "fdsfds",
      "matched": true
    }
  },
  "workerScanId": "worker123",
  "status": "PENDING"
```

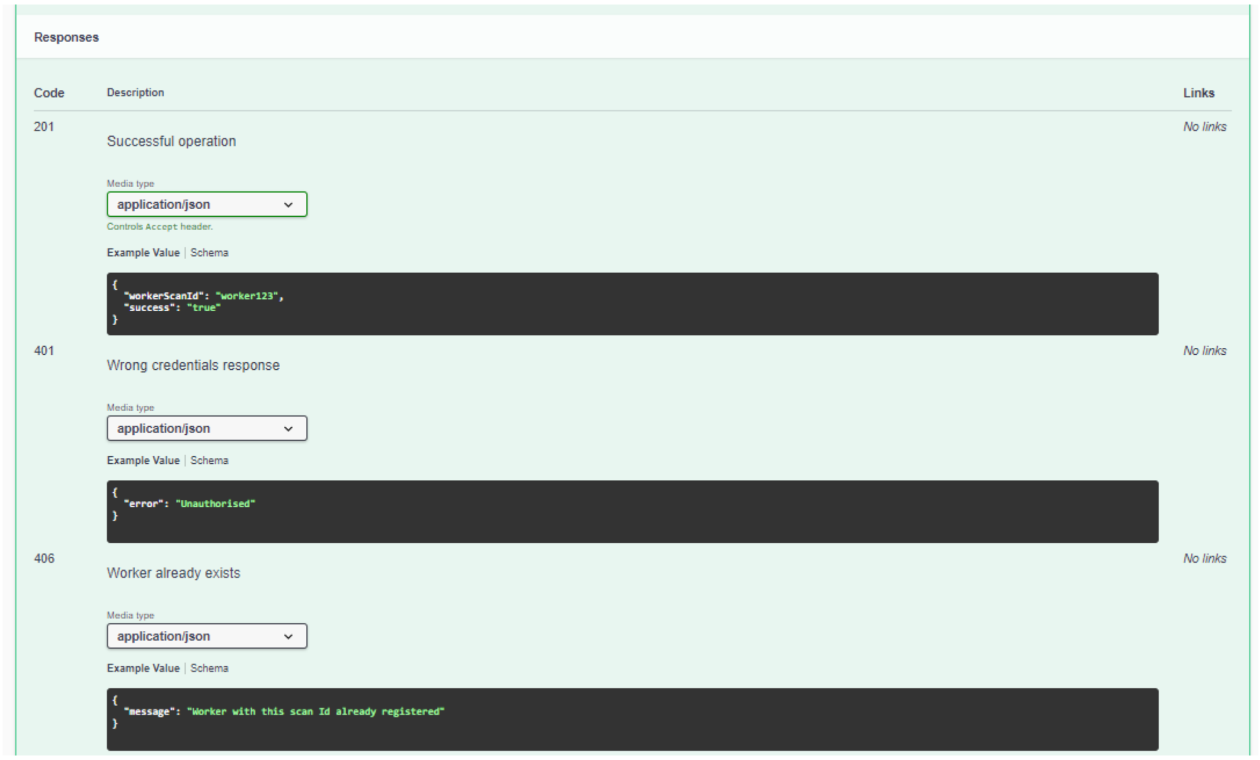
API FOR REGISTERING WORKER OFFLINE

End Point: (baseurl) /CashforWorkLaravel/api/storeOffline
Method: POST
Request Parameters:



Name	Description
Authorization * required string (header)	Bearer token of facilitator Authorization - Bearer token of facilitator
idDocument * required (formData)	Image of Id document idDocument - Image of Id document
selfieDocument * required (formData)	Selfie Image of worker selfieDocument - Selfie Image of worker
manualRegistrationData * required (formData)	Manually entered data of worker *Example* : { "name":"Binod", "mothersName": "Bina", "birthPlace": "Dhading", "dateOfBirth": "hehehe", "gender": "male", "contactNumber": "ghjkjhg" } { "name":"Binod", "mothersName": "Bina", "bi
workerScanId * required (formData)	Scanned id of worker workerScanId - Scanned id of worker

Upon request these are the responses that application gets when the data is manually synced form the app by the user.



Responses

Code	Description	Links
201	Successful operation	*No links*
401	Wrong credentials response	*No links*
406	Worker already exists	*No links*

201 – Media type: application/json (Controls Accept header.) – Example Value | Schema

```
{
  "workerScanId": "worker123",
  "success": "true"
}
```

401 – Media type: application/json – Example Value | Schema

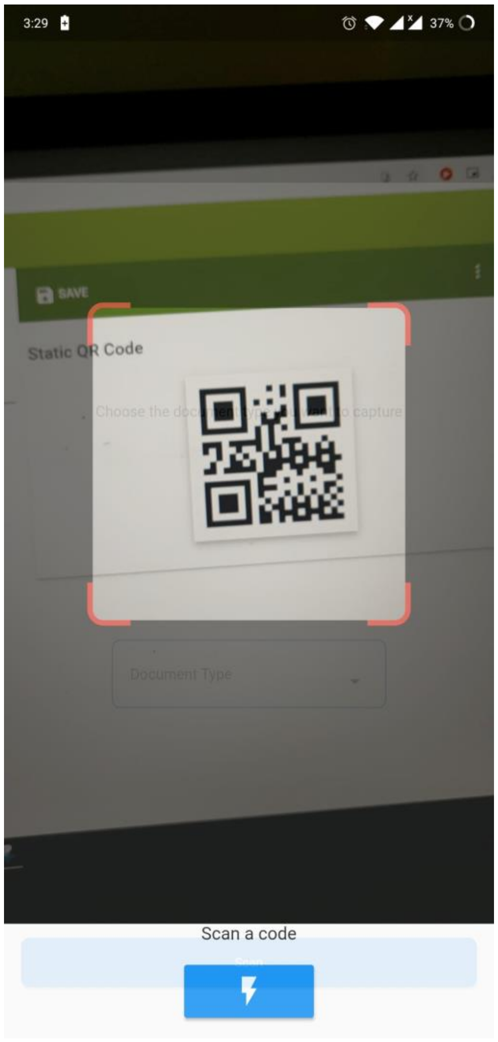
```
{
  "error": "Unauthorised"
}
```

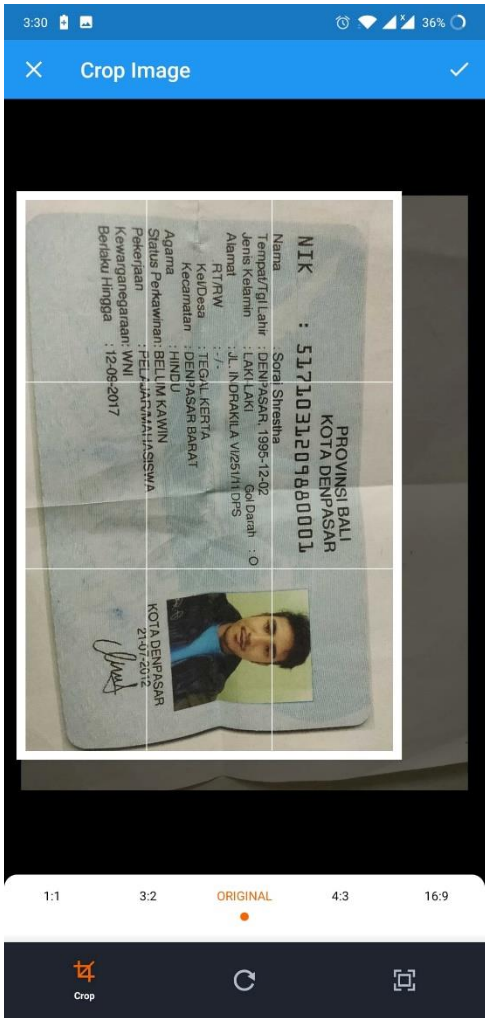
406 – Media type: application/json – Example Value | Schema

```
{
  "message": "Worker with this scan Id already registered"
}
```

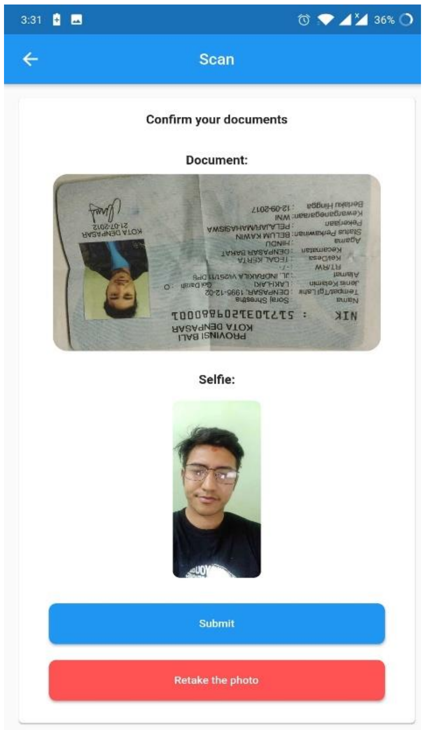
WORKER REGISTRATION ON APPLICATION

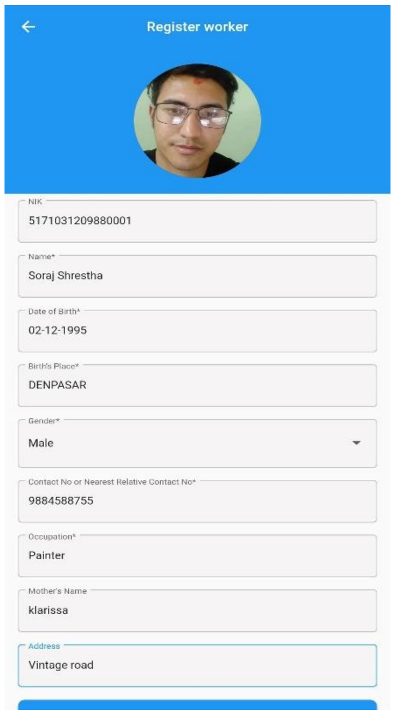
Using the APIs that are listed out above, an interactive worker registration interface has been created on the application. As discussed above for the process of worker registration, same flow has been implemented and can be viewed in the screenshots given below:





PROVINSI BALI
KOTA DENPASAR
NIK : 5171031209880001
Nama : Sorai Shrestha
Tempat/Tgl Lahir : DENPASAR, 1995-12-02
Jenis Kelamin : LAKI-LAKI Gol Darah : O
Alamat : JL. INDRAKILA VI/251/11 DPS
RT/RW : -/-
Kel/Desa : TEGAL KERTA
Kecamatan : DENPASAR BARAT
Agama : HINDU
Status Perkawinan: BELUM KAWIN
Pekerjaan : PELAJAR/MAHASISWA
Kewarganegaraan: WNI
Berlaku Hingga : 12-09-2017
KOTA DENPASAR
21-07-2012





NIK	5171031209880001
Name*	Soraj Shrestha
Date of Birth*	02-12-1995
Birth's Place*	DENPASAR
Gender*	Male
Contact No or Nearest Relative Contact No*	9884588755
Occupation*	Painter
Mother's Name	klarissa
Address	Vintage road

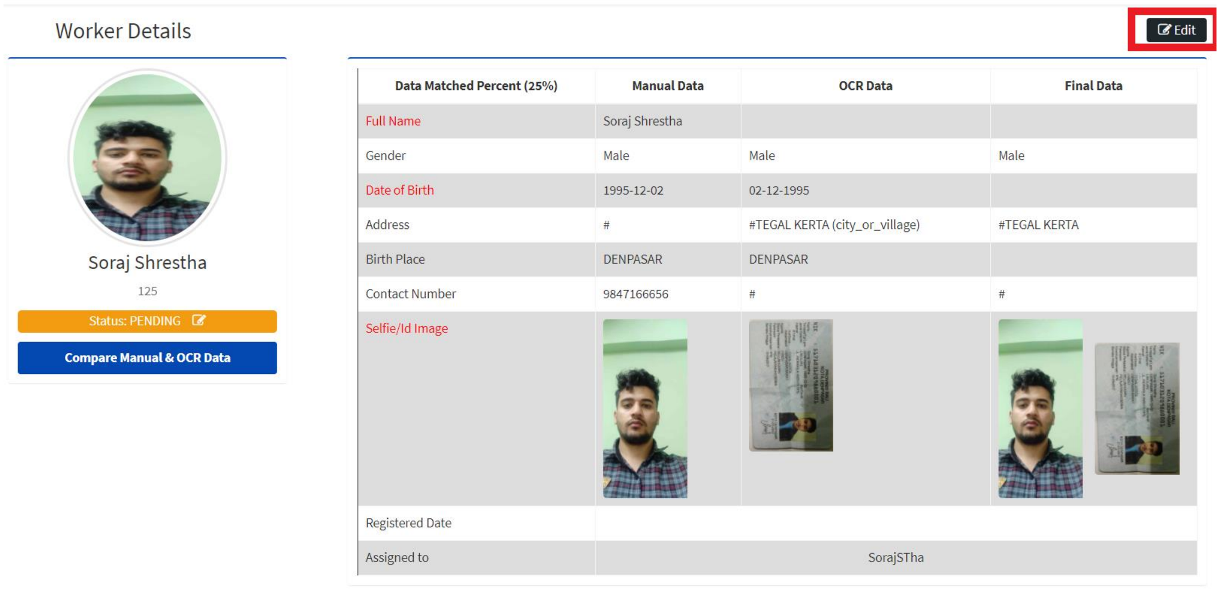
As seen on the screenshots above, the application follow the same flow as discussed earlier on the process section. Detail description of the entire process in the mobile application is also stated below:

- First step of the worker registration is to scan the QR code. This QR code is then assigned to the worker as his/her identity and the same qrcode details are used during attendance and payment process.
- Next is to scan the document. Facilitators have to just take a picture of document and using the cropper tool that shows up on next step they can align the document to meet the requirement.
- After collection of document, a selfie procedure is followed where facilitators will have to take a picture of the worker. This selfie is later on used to verify if the worker is same as shown in document. Also during payment process this selfie can be used to verify if payments are being given to correct personnel.
- Next step is a little different in terms of if there is internet connectivity on the device or not. A form which is to be filled up is displayed.
- During online, if document is clicked properly, the system scans the doc and OCRs the information from it and the form is prefilled with available data.
- But during offline state of app, OCR is not possible so the form is empty and will have to be filled manually.

- The next step is to click on submit and if every step is done correctly the workers are registered to the system.
- Only after workers are successfully registered in the system, they will be able to perform tasks and make checkin/ checkout and payments.

As mentioned earlier as well, after the workers are registed they are classified into various categories based on the match of the document data and manually filled form data. The details can later be changed from the webportal. The process to do so is decribed below:

Suppose we have a worker whose status is Pending as shown below:

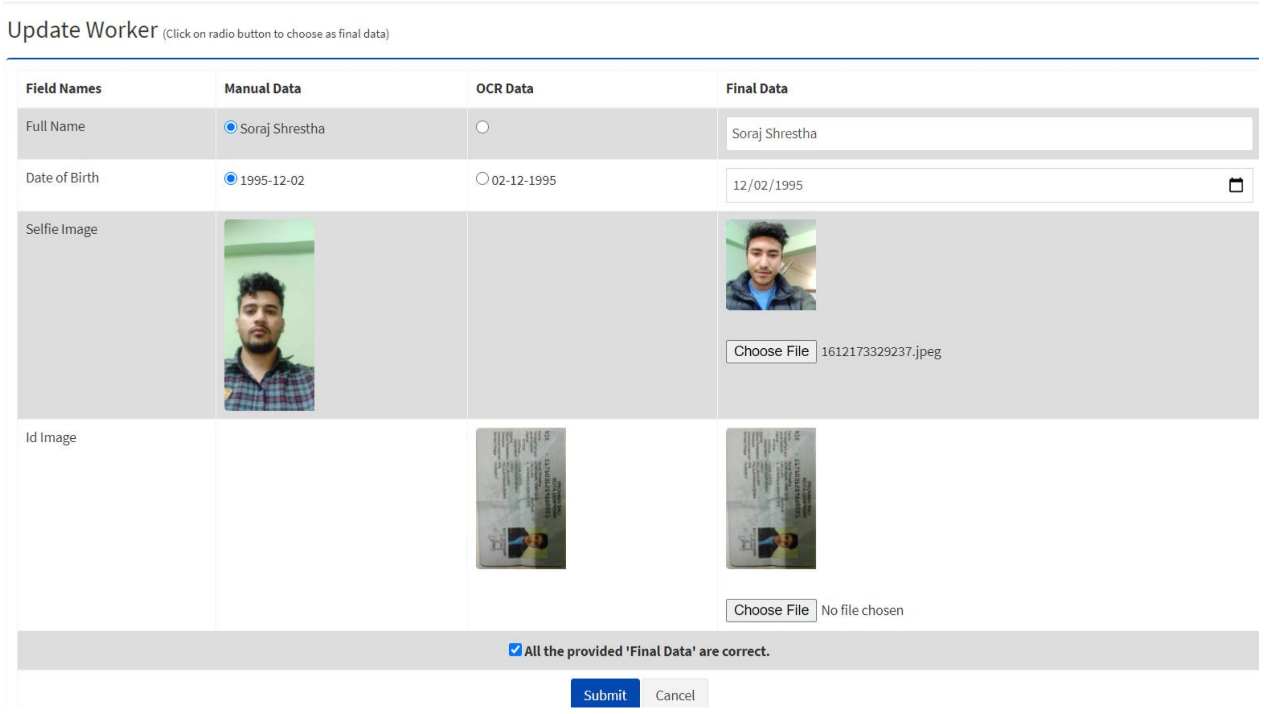


Data Matched Percent (25%)	Manual Data	OCR Data	Final Data
Full Name	Soraj Shrestha		
Gender	Male	Male	Male
Date of Birth	1995-12-02	02-12-1995	
Address	#	#TEGAL KERTA (city_or_village)	#TEGAL KERTA
Birth Place	DENPASAR	DENPASAR	
Contact Number	9847166656	#	#
Selfie/Id Image			
Registered Date			
Assigned to	SorajSTha		

This worker is in pending state as there are fields that are not matching to his OCR data from document. We can see that his name could not be read by OCR system, maybe because clicked image of document is unclear. Date of birth is not matching as well as the worker`s selfie image is not the same as in document.

VERIFYING & FINALIZING WORKER DATA

So to make this worker as verified worker his details need to be changed and finalized. This is done manually by clicking on the edit button as marked in red in the pic above. An interface is created as shown below that allows to edit any unmatched data. The user then inserts and changes data and those data are considered as final data. Once admin/facilitator makes these changes the workers are now verified workers. If everything was matched the worker would have been verified already.


Update Worker (Click on radio button to choose as final data)

Field Names	Manual Data	OCR Data	Final Data
Full Name	● Soraj Shrestha	○	Soraj Shrestha
Date of Birth	● 1995-12-02	○ 02-12-1995	12/02/1995
Selfie Image			Choose File 1612173329237.jpeg
Id Image			Choose File No file chosen

✓ All the provided 'Final Data' are correct.

Picture above the admin/facilitator has made changes/selected correct data and submitted. This manual process now verifies the worker.

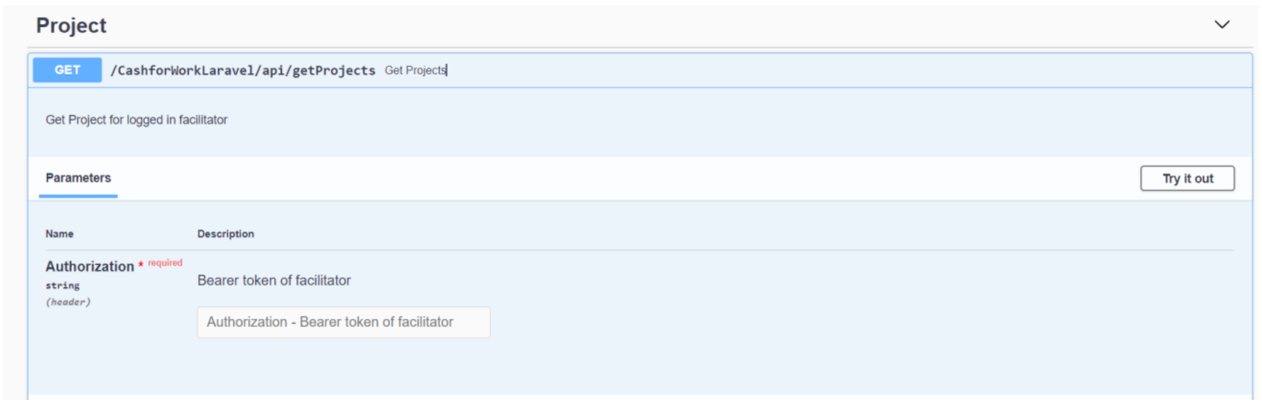
ACTIVITY MODULE

The activity section is where admin create activities on which facilitators are assigned to. As discussed on weekly calls a UI is created to get necessary details to register a activity. A separate module called “Work Type” is also created to assign the type of workers that are required for the activity. Workers will be allowed to select the type of work they will do during check-in based on the selections made here. Rates for each worktype for each day is also declared in this section so the calculation for the payments will be done based on data presented here.

API TO FETCH ACTIVITY

To get list of activities on application, this API manages the request to fetch the activities whenever it is required. Details are as follows:

End Point: (baseurl) /CashforWorkLaravel/api/getProjects
Method: GET
Request Body: Authorization (Bearer Token of facilitator)



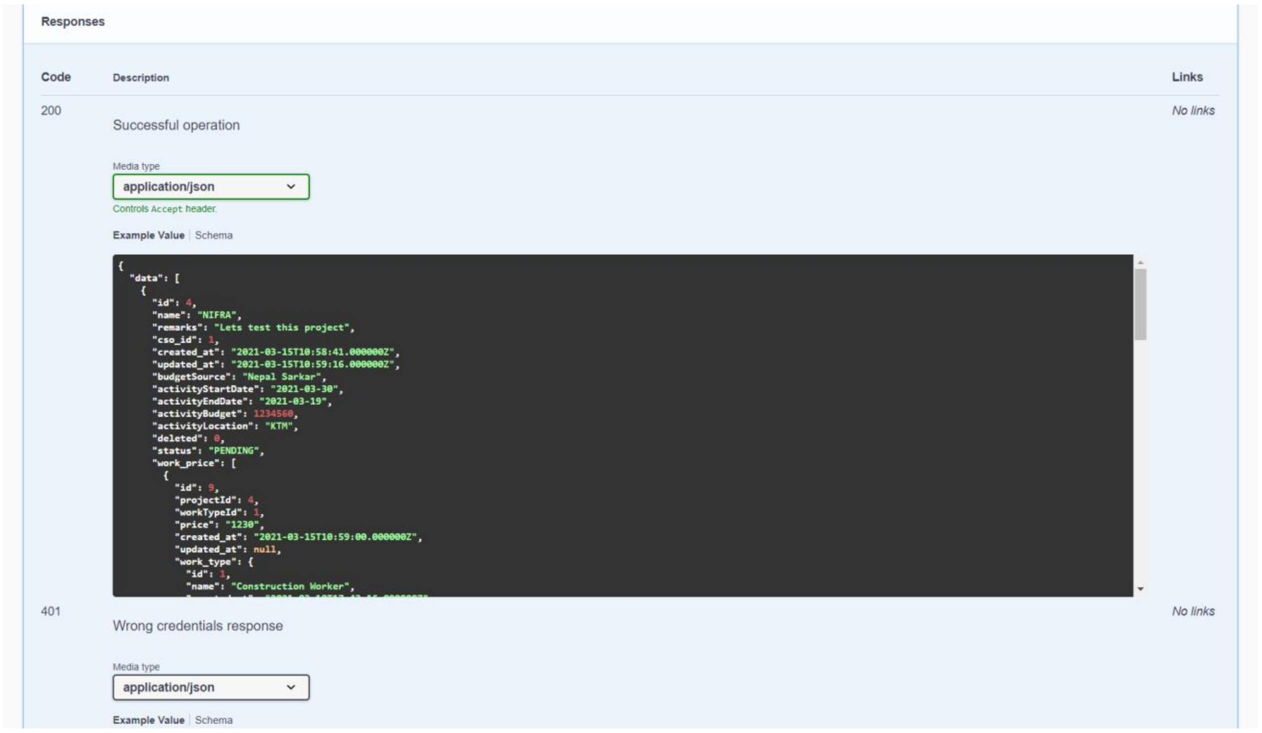
Project

GET `/CashforWorkLaravel/api/getProjects` Get Projects

Get Project for logged in facilitator

Parameters | Try it out

Name	Description
Authorization * required string (header)	Bearer token of facilitator Authorization - Bearer token of facilitator



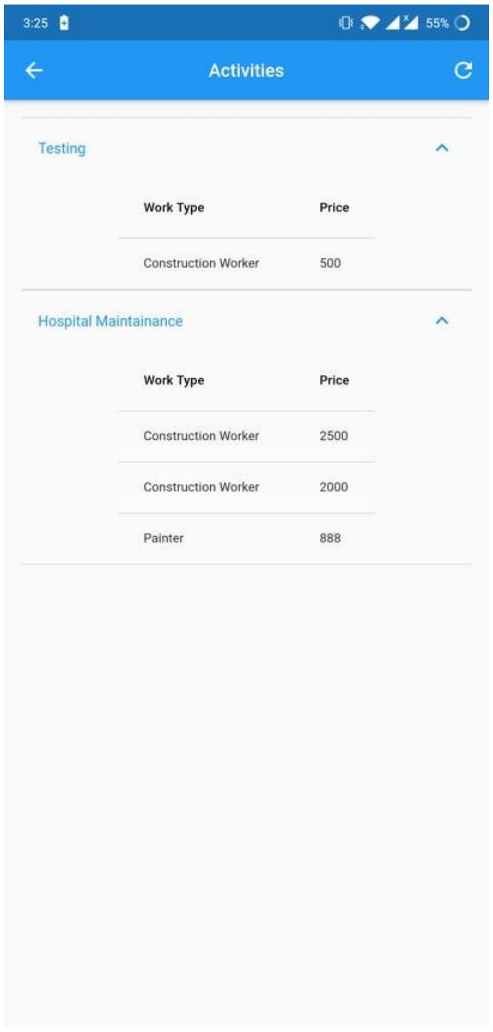
Responses

Code	Description	Links
200	Successful operation Media type: application/json Controls Accept header. Example Value | Schema	No links
401	Wrong credentials response Media type: application/json Example Value | Schema	No links

```
{
  "data": [
    {
      "id": 4,
      "name": "NIFRA",
      "remarks": "Lets test this project",
      "cso_id": 1,
      "created_at": "2021-03-15T10:58:41.000000Z",
      "updated_at": "2021-03-15T10:59:16.000000Z",
      "budgetSource": "Nepal Sarkar",
      "activityStartDate": "2021-03-30",
      "activityEndDate": "2021-03-19",
      "activityBudget": 1234560,
      "activityLocation": "KTM",
      "deleted": 0,
      "status": "PENDING",
      "work_price": [
        {
          "id": 9,
          "projectId": 4,
          "workTypeId": 1,
          "price": "1230",
          "created_at": "2021-03-15T10:59:00.000000Z",
          "updated_at": null,
          "work_type": {
            "id": 1,
            "name": "Construction Worker",
```

Using this API, activities are fetched on the application. By fetching projects and work types associated on the respective activities, workers will be able to select activity and work during check in process. Screenshot of the app below shows the view on app and how projects are used to check in for workers.

MOBILE VIEW OF ACTIVITY AND ITS USE

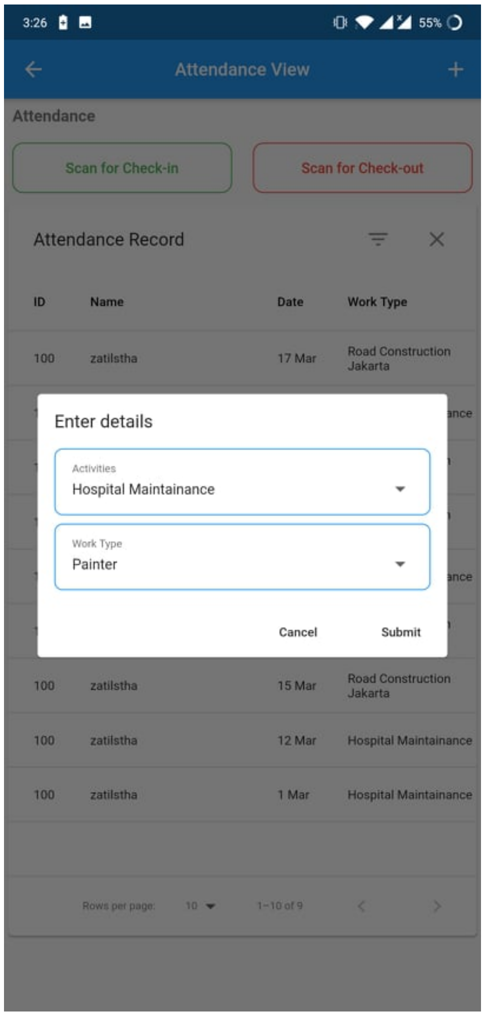


Testing

Work Type	Price
Construction Worker	500

Hospital Maintainance

Work Type	Price
Construction Worker	2500
Construction Worker	2000
Painter	888



ID	Name	Date	Work Type
100	zatilstha	17 Mar	Road Construction Jakarta
100	zatilstha	15 Mar	Road Construction Jakarta
100	zatilstha	12 Mar	Hospital Maintainance
100	zatilstha	1 Mar	Hospital Maintainance

Rows per page: 10 1–10 of 9

Enter details

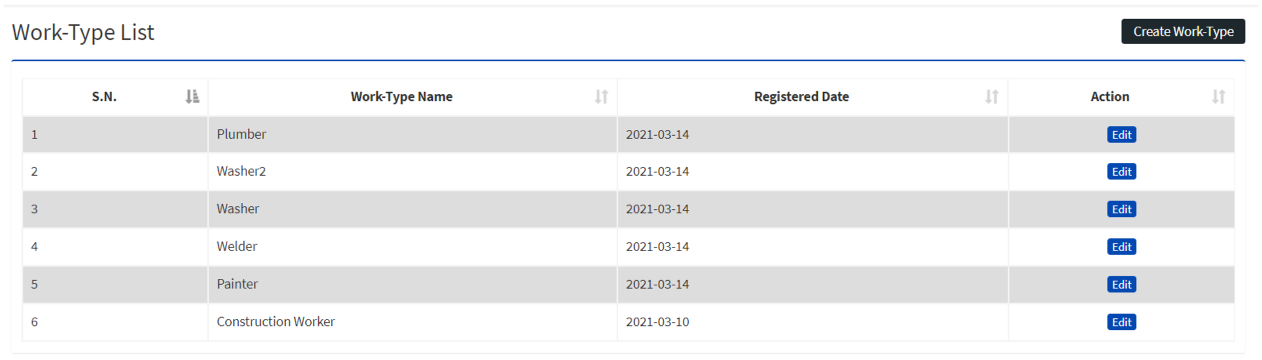
Activities: Hospital Maintainance

Work Type: Painter

ACTIVITY CREATION ON WEB PORTAL

The first step of creating a project would be to first create a list of work types that would be required to conduct the project. There is a separate portal to do so in the web portal and a demonstration of it is shown below.

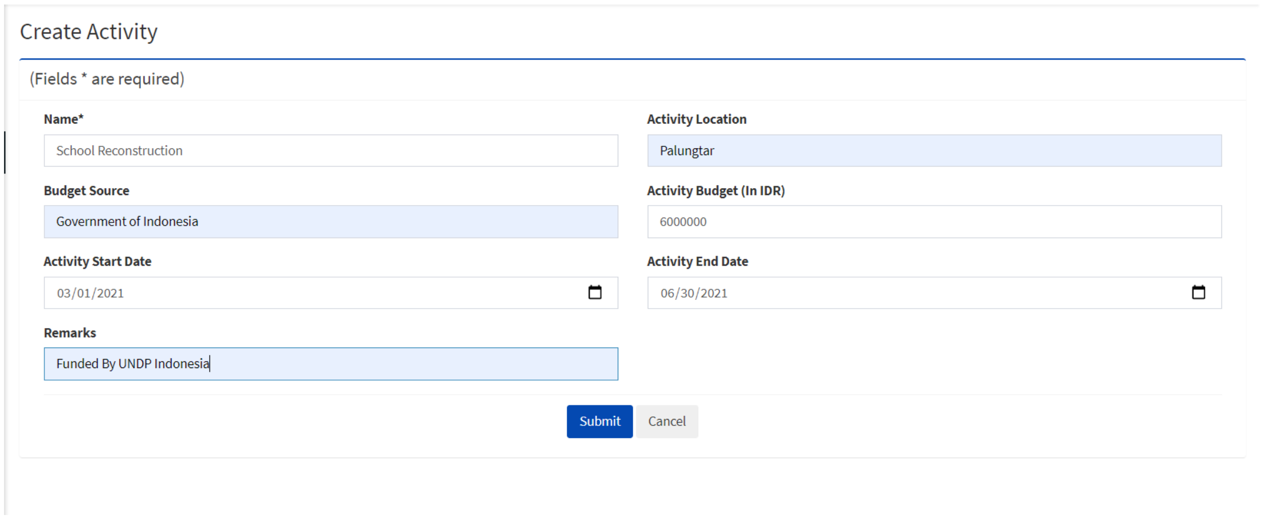
At the very top there is a create work type button. By clicking on it and entering the name of the work type we can list out as many different work types as necessary to conduct the project.



Work-Type List

S.N.	Work-Type Name	Registered Date	Action
1	Plumber	2021-03-14	Edit
2	Washer2	2021-03-14	Edit
3	Washer	2021-03-14	Edit
4	Welder	2021-03-14	Edit
5	Painter	2021-03-14	Edit
6	Construction Worker	2021-03-10	Edit

Once the necessary worktypes are listed, we can then proceed to creating activities. There are three basic forms to be filled up during activity creation. First would be Basic information such as activity name, budget, location, budget souce and activity duration. Screenshot below displays the necessary and optional fields required to fill up for this step.

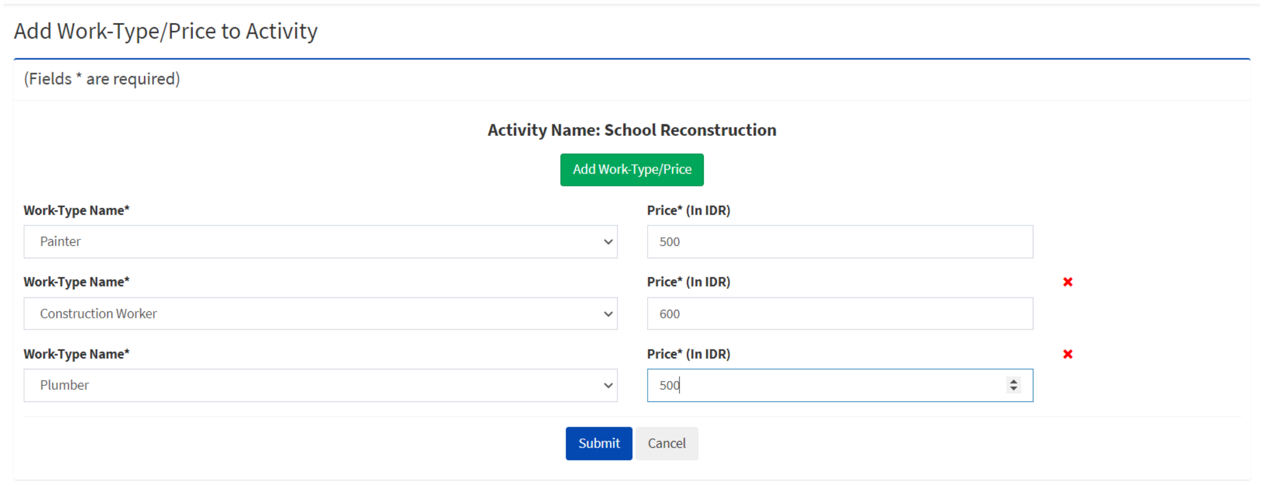


Create Activity

(Fields * are required)

Name*	School Reconstruction
Activity Location	Palungtar
Budget Source	Government of Indonesia
Activity Budget (In IDR)	6000000
Activity Start Date	03/01/2021
Activity End Date	06/30/2021
Remarks	Funded By UNDP Indonesia

Next step would be to add worktype and rates for each worktype. The worktype we created in the worktype section earlier will be listed here in the drop down. To add more than one worktype, there is a add button. Corresponding to the worktype we can set the rate of it that will be used to calculate payment of the worker.

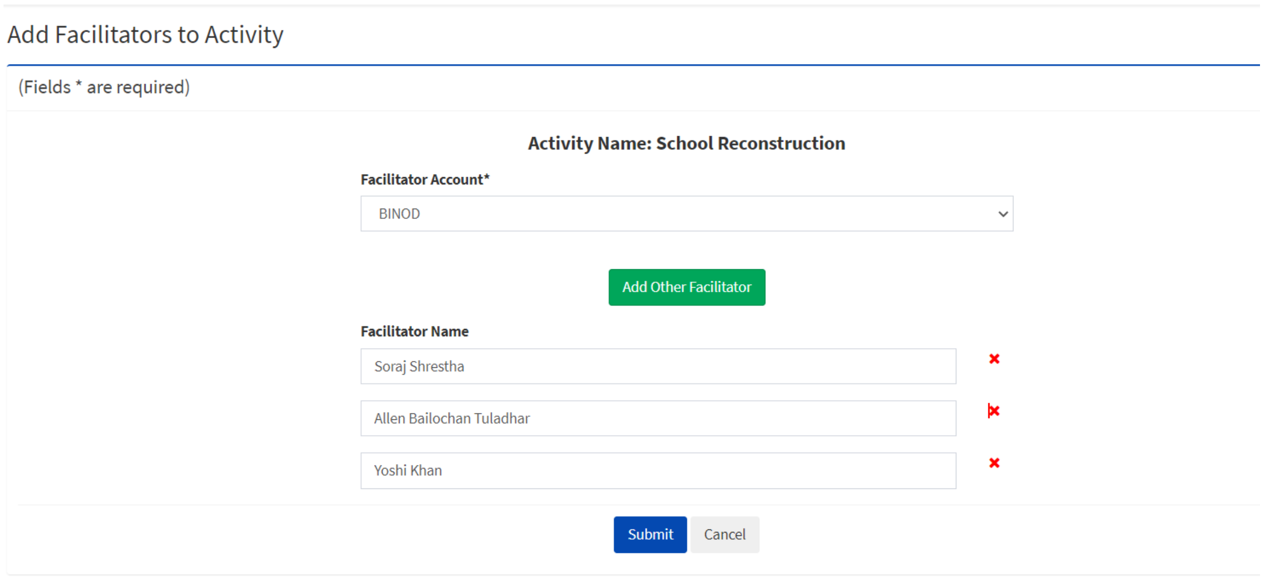

Add Work-Type/Price to Activity

(Fields * are required)

Activity Name: School Reconstruction

Work-Type Name*	Price* (In IDR)
Painter	500
Construction Worker	600
Plumber	500

Finally we will now be directed to another page where we will have to assign facillitators. For each activity there will be only one facilitaor account and each of the facilitator will use this account but we can enlist other facilitators as well using the add button. The other facilitators added manually will not have any login account in the application and will have to use the facilitator account selected here.


Add Facilitators to Activity

(Fields * are required)

Activity Name: School Reconstruction

Facilitator Account*: BINOD

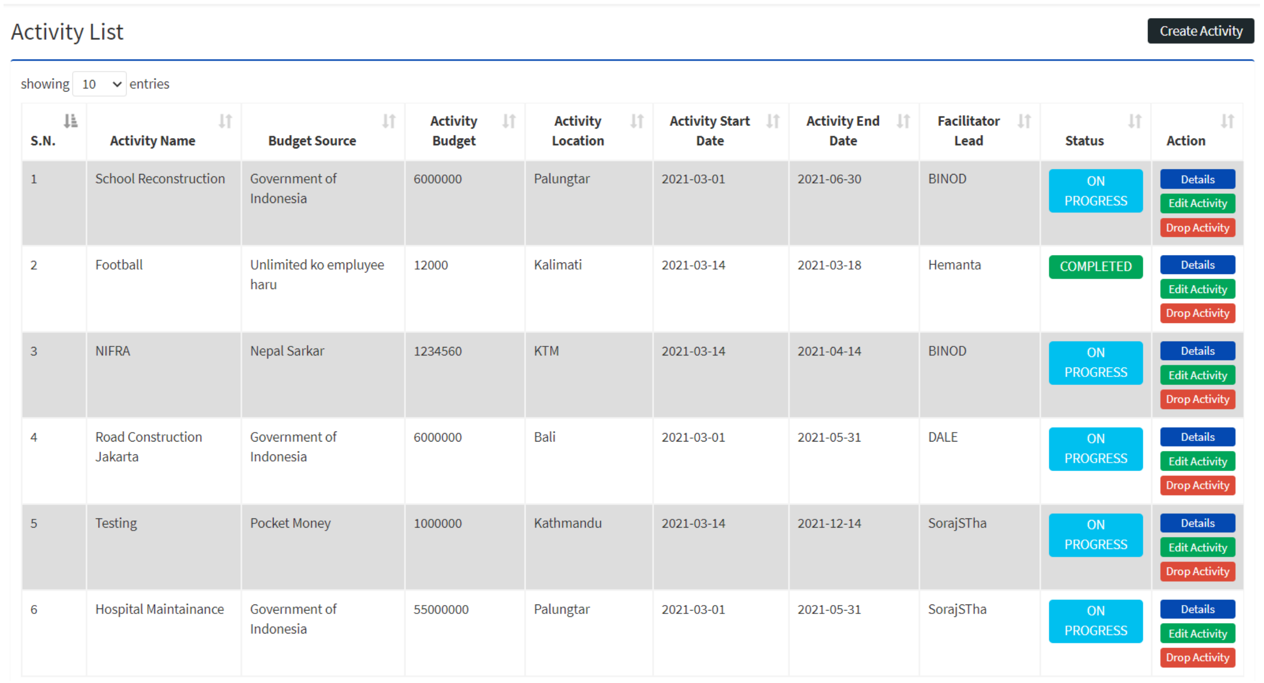
Facilitator Name

Soraj Shrestha

Allen Bailochan Tuladhar

Yoshi Khan

Performing all these steps correctly we will be able to create activities. Once an activity is created it is listed in the activities section landing page with basic details and staus of it. There are other action buttons as well that has various functionalites that is self explanatory.

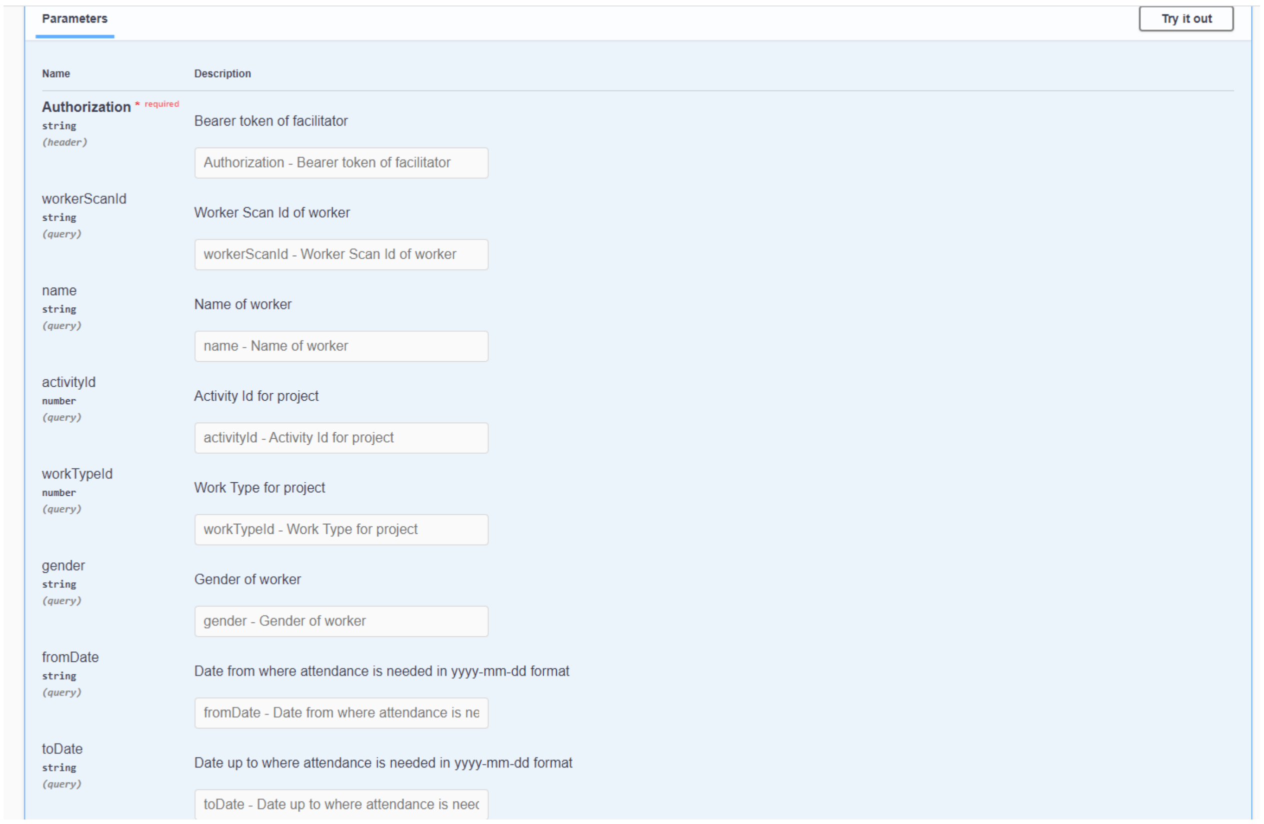

Activity List

S.N.	Activity Name	Budget Source	Activity Budget	Activity Location	Activity Start Date	Activity End Date	Facilitator Lead	Status	Action
1	School Reconstruction	Government of Indonesia	6000000	Palungtar	2021-03-01	2021-06-30	BINOD	ON PROGRESS	Details Edit Activity Drop Activity
2	Football	Unlimited ko employee haru	12000	Kalimati	2021-03-14	2021-03-18	Hemanta	COMPLETED	Details Edit Activity Drop Activity
3	NIFRA	Nepal Sarkar	1234560	KTM	2021-03-14	2021-04-14	BINOD	ON PROGRESS	Details Edit Activity Drop Activity
4	Road Construction Jakarta	Government of Indonesia	6000000	Bali	2021-03-01	2021-05-31	DALE	ON PROGRESS	Details Edit Activity Drop Activity
5	Testing	Pocket Money	1000000	Kathmandu	2021-03-14	2021-12-14	SorajSTha	ON PROGRESS	Details Edit Activity Drop Activity
6	Hospital Maintainance	Government of Indonesia	55000000	Palungtar	2021-03-01	2021-05-31	SorajSTha	ON PROGRESS	Details Edit Activity Drop Activity

ATTENDANCE

API FOR FETCHING ATTENDANCE LIST

To get list of attendance on application and webportal, this API manages the request to fetch the attendance with required filters required. The API is designed in such a way that when user requests for attendance list by passing certain filters, it returns result based on it. For example if user wants attendance of a particular date or of a certain gender or maybe any other filter this API collects the requirement and sends the attendance list accordingly. Details are as follows:
End Point: (baseurl) /CashforWorkLaravel/api/attendance
Method: GET
Request Body: Authorization (Bearer Token of facilitator)
Parameters: (shown in pic below)

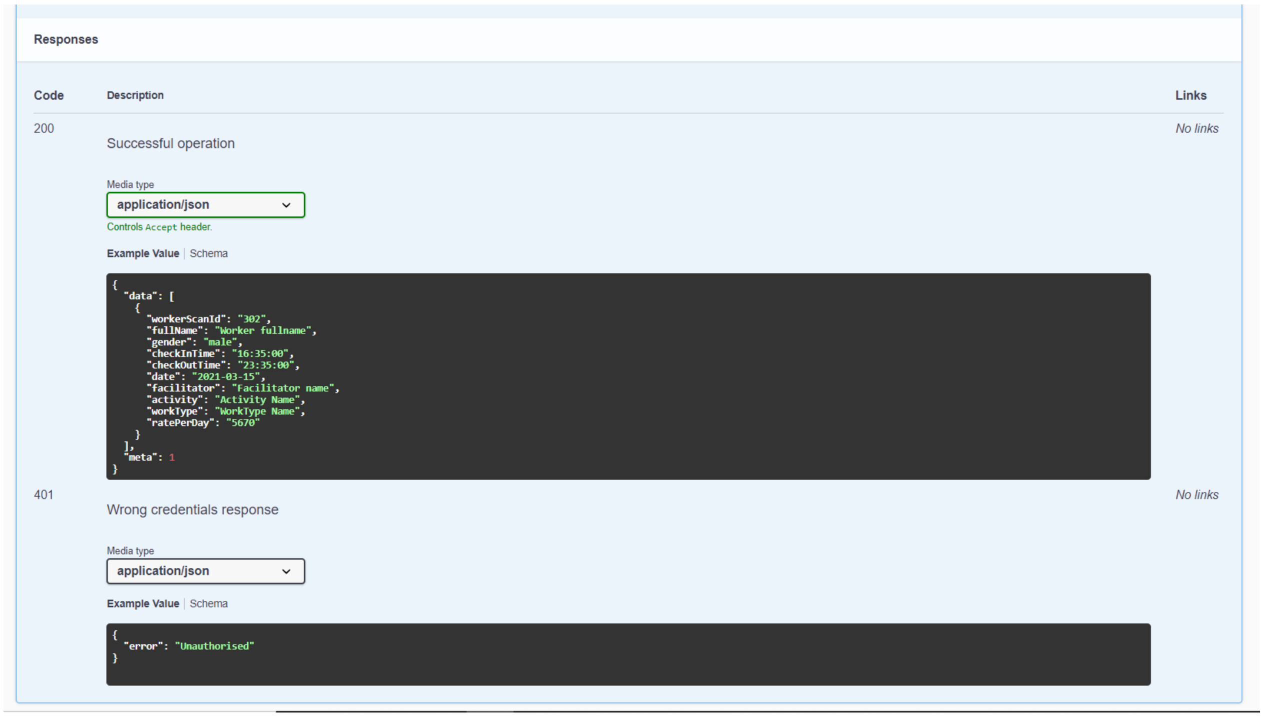


Parameters

Try it out

Name	Description
Authorization * required string *(header)*	Bearer token of facilitator Authorization - Bearer token of facilitator
workerScanId string *(query)*	Worker Scan Id of worker workerScanId - Worker Scan Id of worker
name string *(query)*	Name of worker name - Name of worker
activityId number *(query)*	Activity Id for project activityId - Activity Id for project
workTypeId number *(query)*	Work Type for project workTypeId - Work Type for project
gender string *(query)*	Gender of worker gender - Gender of worker
fromDate string *(query)*	Date from where attendance is needed in yyyy-mm-dd format fromDate - Date from where attendance is ne
toDate string *(query)*	Date up to where attendance is needed in yyyy-mm-dd format toDate - Date up to where attendance is neec

If any parameter is not sent then it sends list of all the attendance. But usually when facilitator requests for his/her workers attendance token is passed and only attendance of their attendance is sent. The response received is as shown below:



Responses

Code	Description	Links
200	Successful operation	*No links*
401	Wrong credentials response	*No links*

200 — Media type: application/json
Controls `Accept` header.

Example Value | Schema

```
{
  "data": [
    {
      "workerScanId": "302",
      "fullName": "Worker fullname",
      "gender": "male",
      "checkInTime": "16:35:00",
      "checkOutTime": "23:35:00",
      "date": "2021-03-15",
      "facilitator": "Facilitator name",
      "activity": "Activity Name",
      "workType": "WorkType Name",
      "ratePerDay": "5670"
    }
  ],
  "meta": 1
}
```

401 — Media type: application/json

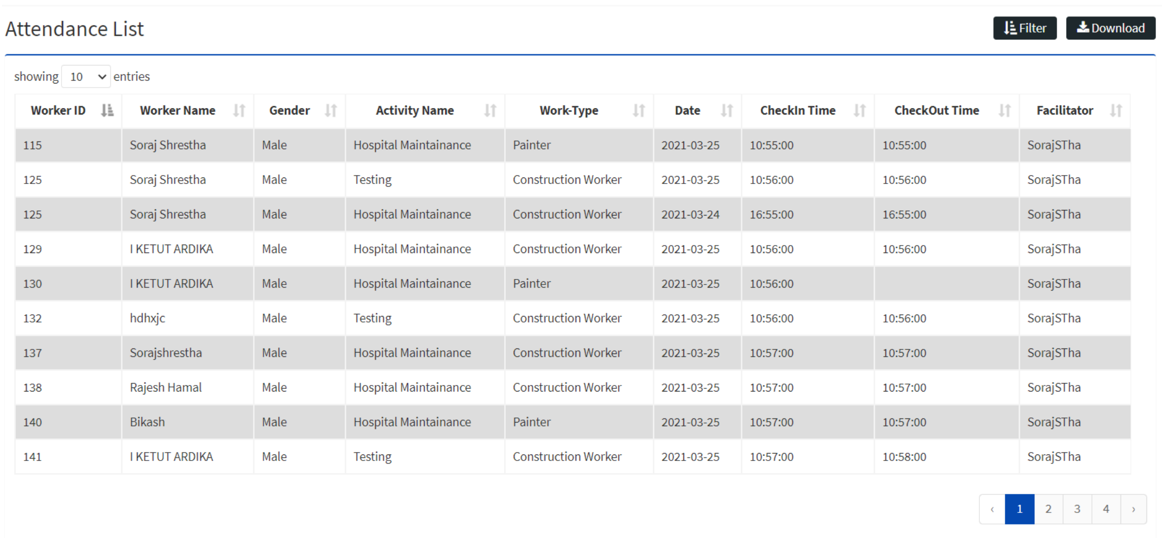
Example Value | Schema

```
{
  "error": "Unauthorised"
}
```

VIEWS FOR ATTENDANCE

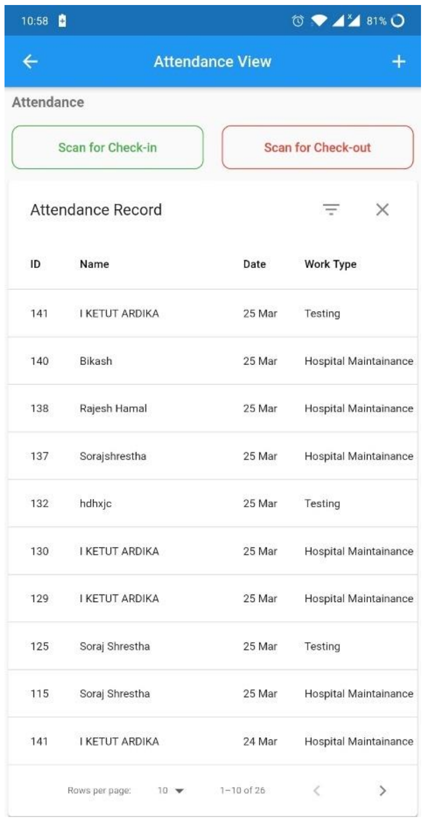
The attendance module is designed in a way that displays attendance from recent date to earlier ones. Each facilitator will be displayed attendance of their concerned workers. If no any filter value is sent all the attendance can be viewed. If we want attendance of specific detail then we can use the filter option given in both app and web portal.

The screenshot below demonstrates the different views of the attendance model



Attendance List

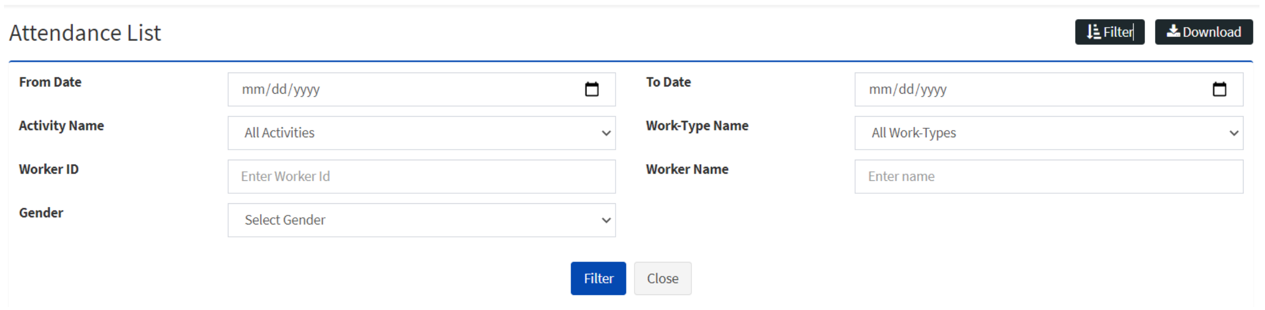
Worker ID	Worker Name	Gender	Activity Name	Work-Type	Date	CheckIn Time	CheckOut Time	Facilitator
115	Soraj Shrestha	Male	Hospital Maintainance	Painter	2021-03-25	10:55:00	10:55:00	SorajSTha
125	Soraj Shrestha	Male	Testing	Construction Worker	2021-03-25	10:56:00	10:56:00	SorajSTha
125	Soraj Shrestha	Male	Hospital Maintainance	Construction Worker	2021-03-24	16:55:00	16:55:00	SorajSTha
129	I KETUT ARDIKA	Male	Hospital Maintainance	Construction Worker	2021-03-25	10:56:00	10:56:00	SorajSTha
130	I KETUT ARDIKA	Male	Hospital Maintainance	Painter	2021-03-25	10:56:00		SorajSTha
132	hdhxjc	Male	Testing	Construction Worker	2021-03-25	10:56:00	10:56:00	SorajSTha
137	Sorajshrestha	Male	Hospital Maintainance	Construction Worker	2021-03-25	10:57:00	10:57:00	SorajSTha
138	Rajesh Hamal	Male	Hospital Maintainance	Construction Worker	2021-03-25	10:57:00	10:57:00	SorajSTha
140	Bikash	Male	Hospital Maintainance	Painter	2021-03-25	10:57:00	10:57:00	SorajSTha
141	I KETUT ARDIKA	Male	Testing	Construction Worker	2021-03-25	10:57:00	10:58:00	SorajSTha



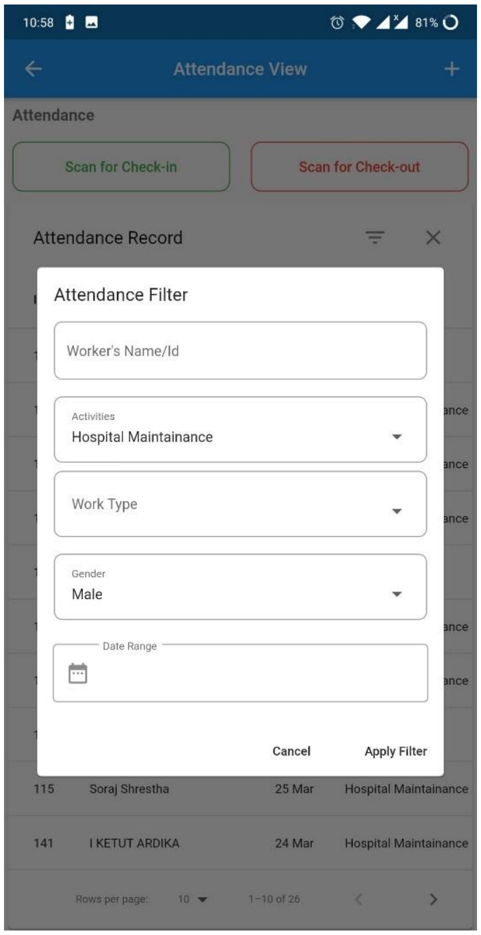
ID	Name	Date	Work Type
141	I KETUT ARDIKA	25 Mar	Testing
140	Bikash	25 Mar	Hospital Maintainance
138	Rajesh Hamal	25 Mar	Hospital Maintainance
137	Sorajshrestha	25 Mar	Hospital Maintainance
132	hdhxjc	25 Mar	Testing
130	I KETUT ARDIKA	25 Mar	Hospital Maintainance
129	I KETUT ARDIKA	25 Mar	Hospital Maintainance
125	Soraj Shrestha	25 Mar	Testing
115	Soraj Shrestha	25 Mar	Hospital Maintainance
141	I KETUT ARDIKA	24 Mar	Hospital Maintainance

The two pictures are an example of what facilitator/admin could view when they click on the attendance tab. Initially, all of his/her worker`s attendances are listed by recent date first order. On the app there is a Check-in and Check-out button as well to make attendance of the worker. To filter out this attendance list there is filter button at the top of the screen. There are various fields from which we can filter out the data which is shown in the sceenshot below:

FILTERING DATA


Attendance List

From Date	mm/dd/yyyy	To Date	mm/dd/yyyy
Activity Name	All Activities	Work-Type Name	All Work-Types
Worker ID	Enter Worker Id	Worker Name	Enter name
Gender	Select Gender		



115	Soraj Shrestha	25 Mar	Hospital Maintainance
141	I KETUT ARDIKA	24 Mar	Hospital Maintainance

As seen on the screenshot, we can filter the data using parameters such as:

* Attendance of a single worker using his/her id or name. This list will be filtered out to display only selected persons data.

*Attendance made on a particular activity. This will filter data to display list of attendance on selected activity.

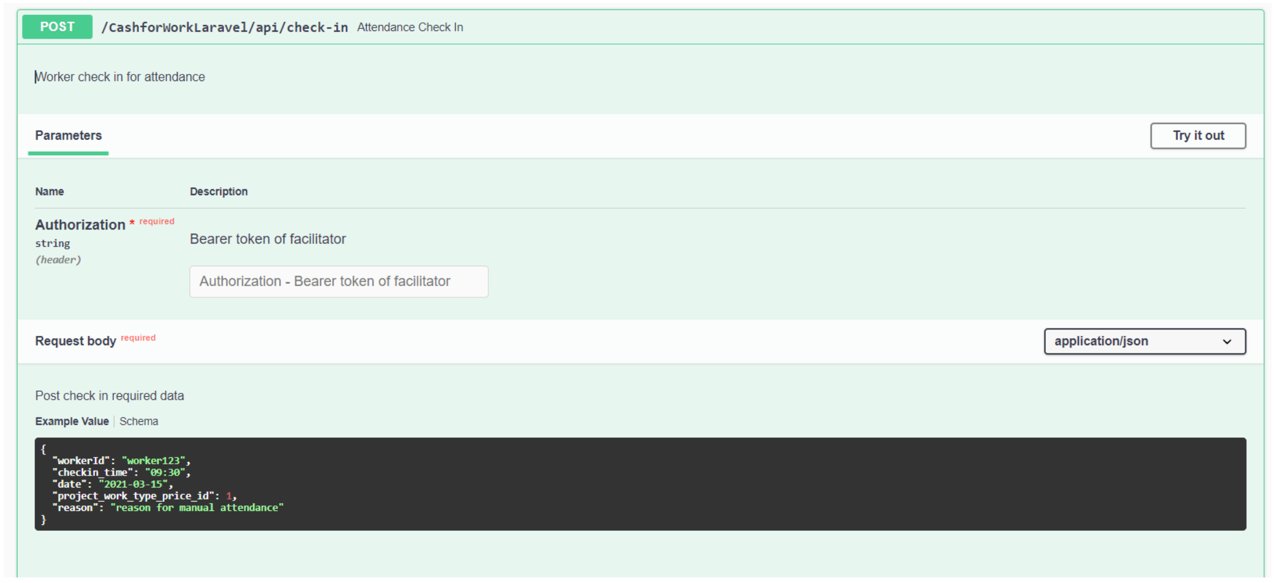
*Attendance of a certain work group type

*Attendance list of specific gender group

*Attendance of a range of days by selecting date.

CHECKING-IN AND CHECKING-OUT

The API used to make attendance check-in is shown below:
End Point: (baseurl) /CashforWorkLaravel/api/check-in
Method: POST
Request Body: Authorization (Bearer Token of facilitator)
Parameters: Worker ID, check-in time, date, project, worktype, reason (in case of manual attendance)


POST /CashforWorkLaravel/api/check-in Attendance Check In

Worker check in for attendance

Parameters

Try it out

Name	Description
Authorization * required string (header)	Bearer token of facilitator Authorization - Bearer token of facilitator

Request body required

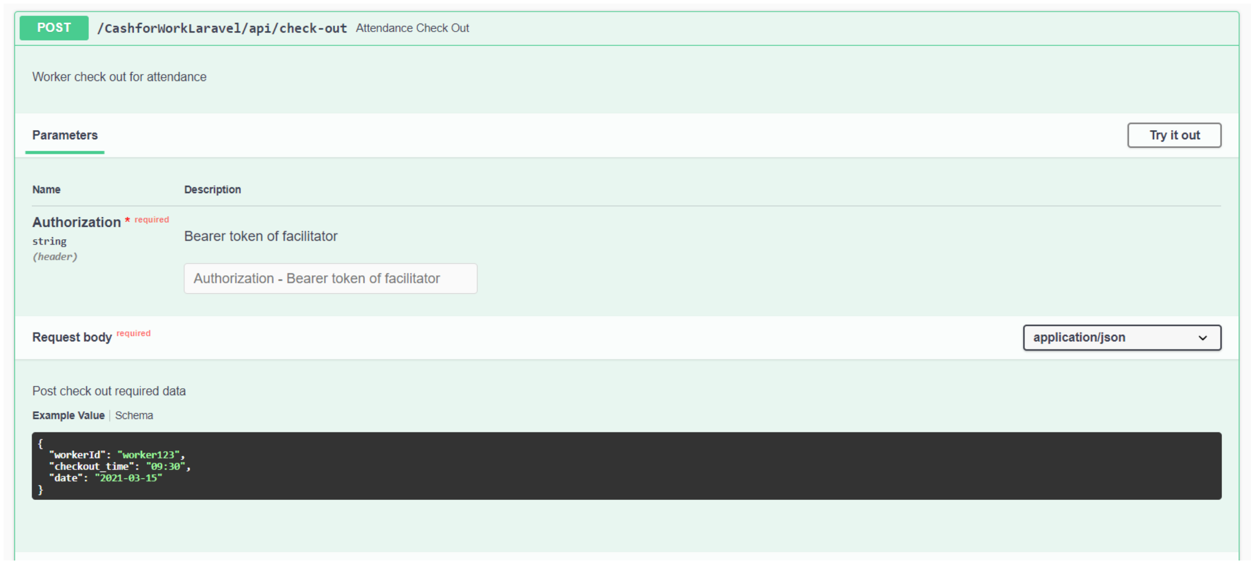
application/json

Post check in required data

Example Value | Schema

```
{
  "workerId": "worker123",
  "checkin_time": "09:30",
  "date": "2021-03-15",
  "project_work_type_price_id": 1,
  "reason": "reason for manual attendance"
}
```

Similarly, for checkout,
The API used to make attendance check-out is shown below:
End Point: (baseurl) /CashforWorkLaravel/api/check-out
Method: POST
Request Body: Authorization (Bearer Token of facilitator)
Parameters: Worker ID, check-out time, date, reason (in case of manual attendance)



POST /CashforWorkLaravel/api/check-out Attendance Check Out

Worker check out for attendance

Parameters

Try it out

Name	Description
Authorization * required string (header)	Bearer token of facilitator Authorization - Bearer token of facilitator

Request body required

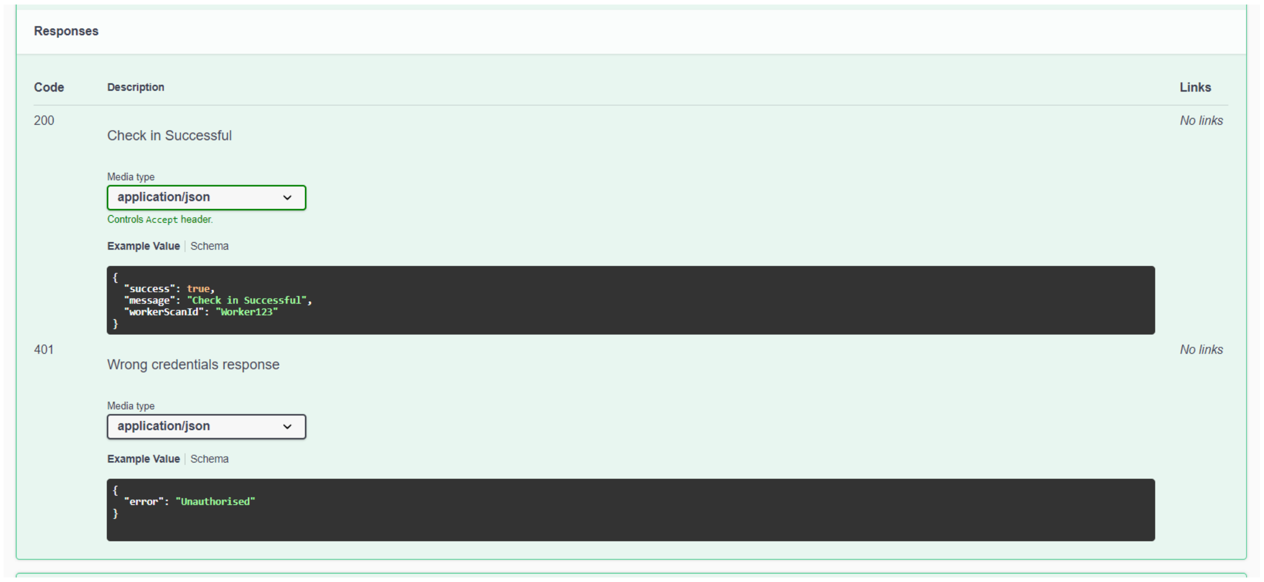
application/json

Post check out required data

Example Value | Schema

```
{
  "workerId": "worker123",
  "checkout_time": "09:30",
  "date": "2021-03-15"
}
```

The response we receive on successful API hit is shown below:



Responses

Code	Description	Links
200	Check in Successful	No links
401	Wrong credentials response	No links

200 — Media type: application/json

Controls Accept header.

Example Value | Schema

```
{
  "success": true,
  "message": "Check in Successful",
  "workerScanId": "Worker123"
}
```

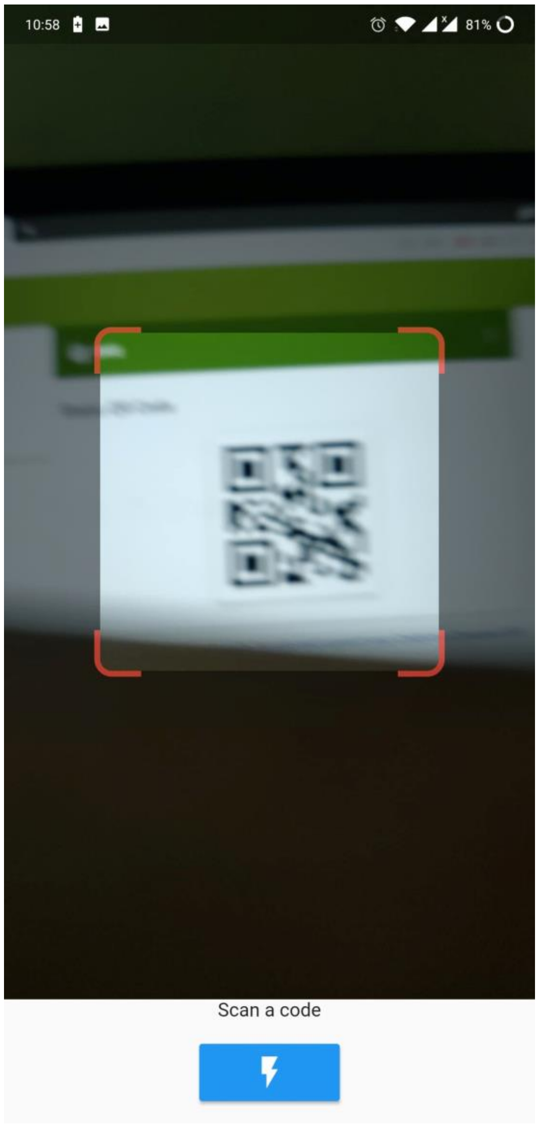
401 — Media type: application/json

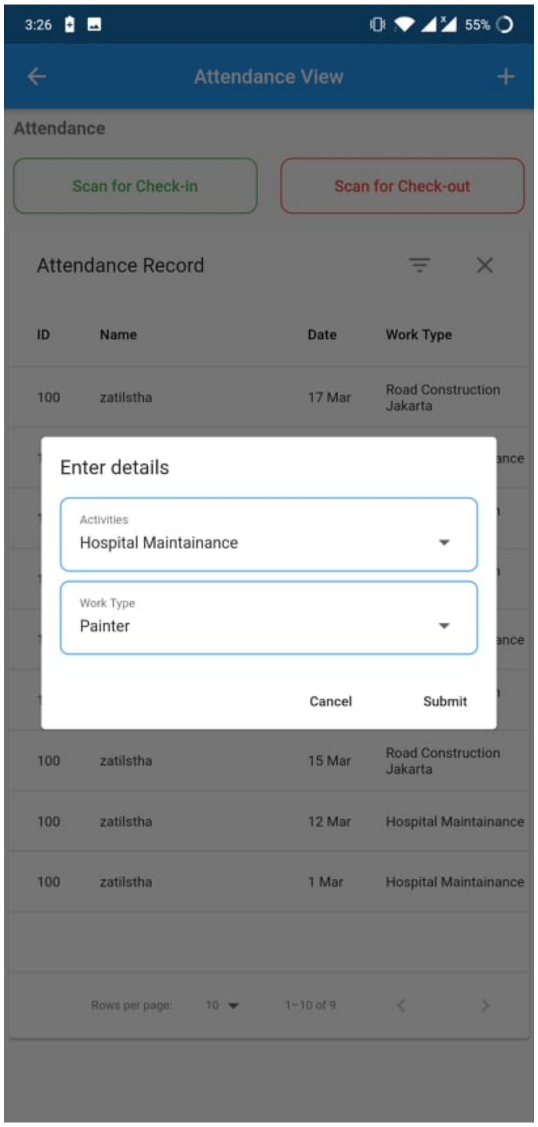
Example Value | Schema

```
{
  "error": "Unauthorised"
}
```

UI TO MAKE ATTENDANCE

To make the attendance, there is a check-in and checkout button on the top of the attendance page. When we click on check-in or check-out, we are redirected to QR scanner that will scan the unique id of worker whose attendance is to be made. During check-in process we need to populate other workers details such as the activity and work type, they will be working on. These details are however not required during checkout and as soon as QR scan is made the checkout process is completed. Images shown are demonstration of the process to make attendance:





ID	Name	Date	Work Type
100	zatilstha	17 Mar	Road Construction Jakarta
100	zatilstha	15 Mar	Road Construction Jakarta
100	zatilstha	12 Mar	Hospital Maintainance
100	zatilstha	1 Mar	Hospital Maintainance

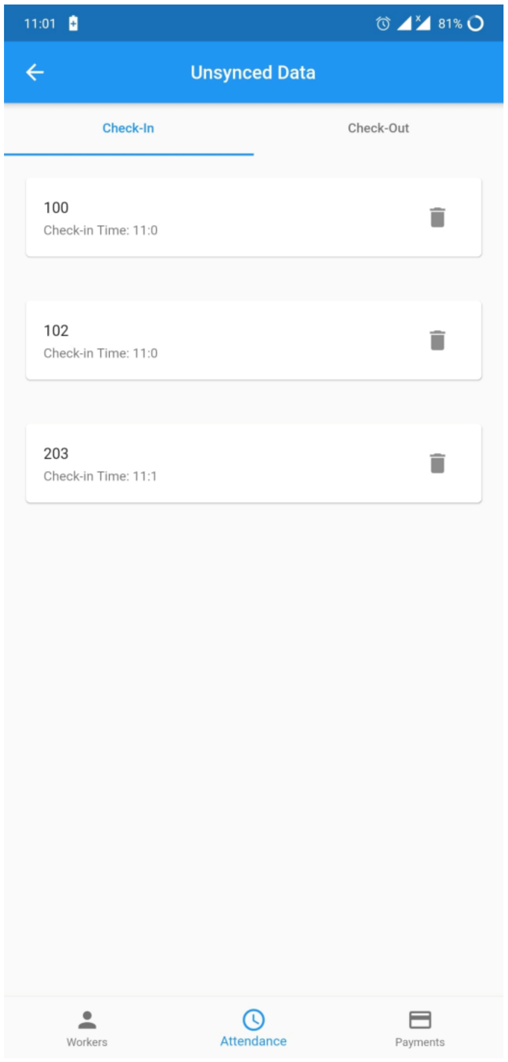
Rows per page: 10 1–10 of 9

Enter details

Activities: Hospital Maintainance

Work Type: Painter

If the application is in online state then attendance data is synced in to the system but if in case there is no internet connectivity then these data are stored in local database and we will have to sync the data to send it to the system. We can view our unsynced data by clicking the chart on the home screen. Screenshot below is an example of unsynced data on our device.



100
Check-in Time: 11:0

102
Check-in Time: 11:0

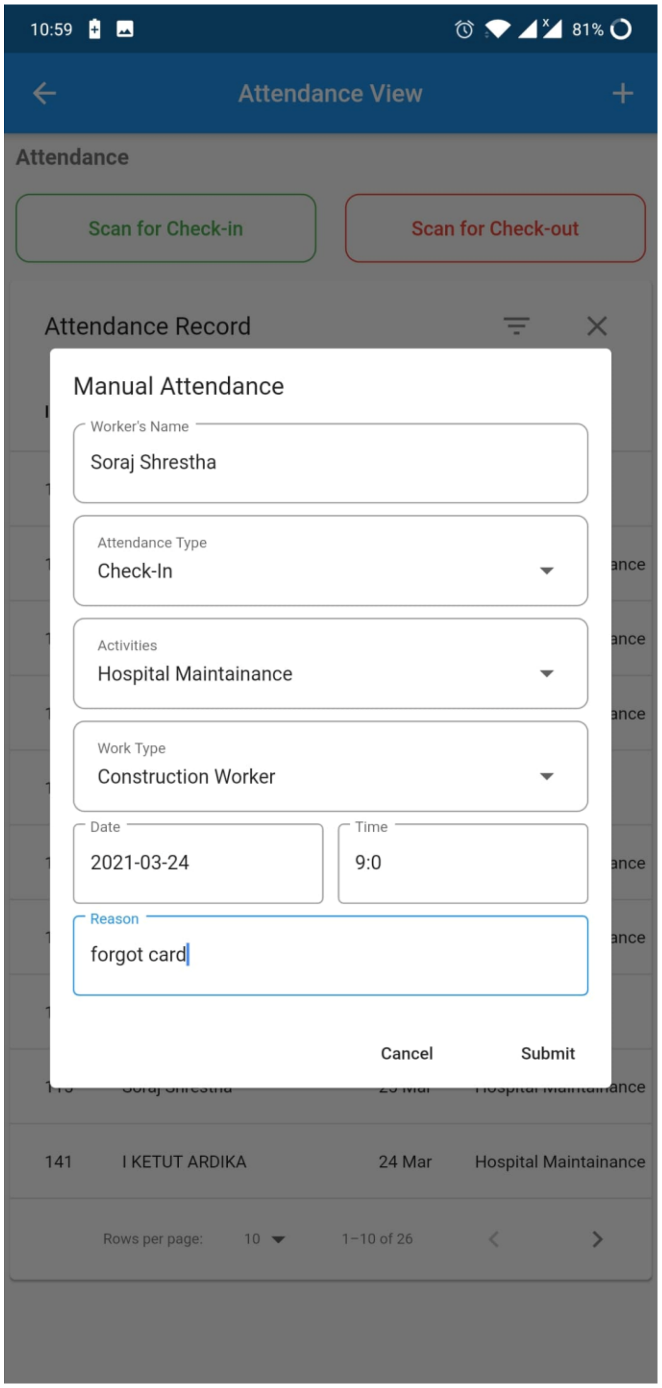
203
Check-in Time: 11:1

MANUAL ATTENDANCE

The application also features a manual attendance system where if in case any workers cannot make attandance because of various reasons. Maybe they did not bring card, or forgot to check in/ out, then facilitators can make their attendance manually using the system. The process to do so is:

- First find the worker using his name or id
- Select if it is a check in or checkout information
- Incase of check in additional data such as activity and worktype will have to be entered.
- Finally enter the date and Check-in or Check-out time

Image Below is a UI of Manual attendance:



Manual Attendance

Worker's Name: Soraj Shrestha

Attendance Type: Check-In

Activities: Hospital Maintainance

Work Type: Construction Worker

Date: 2021-03-24

Time: 9:0

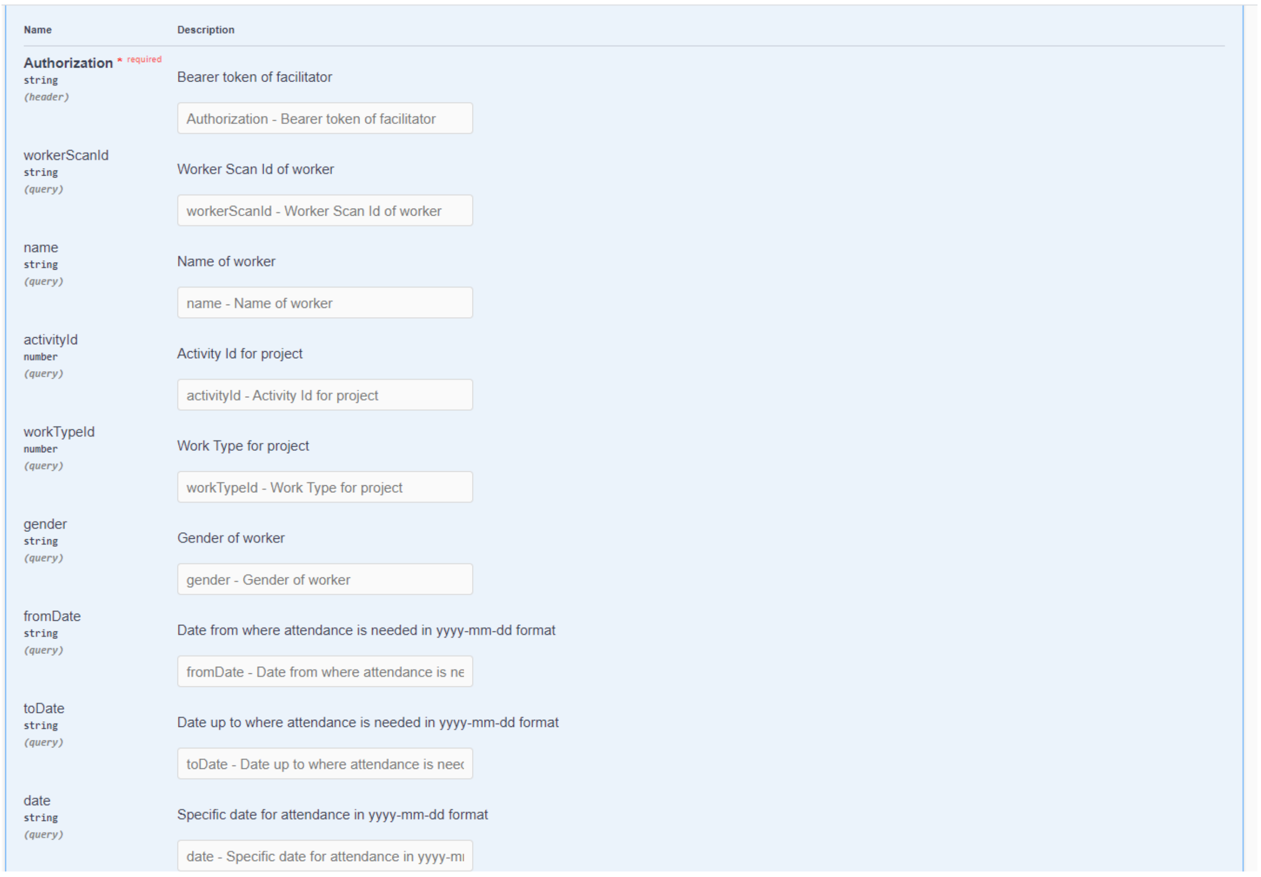
Reason: forgot card

141 I KETUT ARDIKA 24 Mar Hospital Maintainance

GENERATING ATTENDANCE REPORT

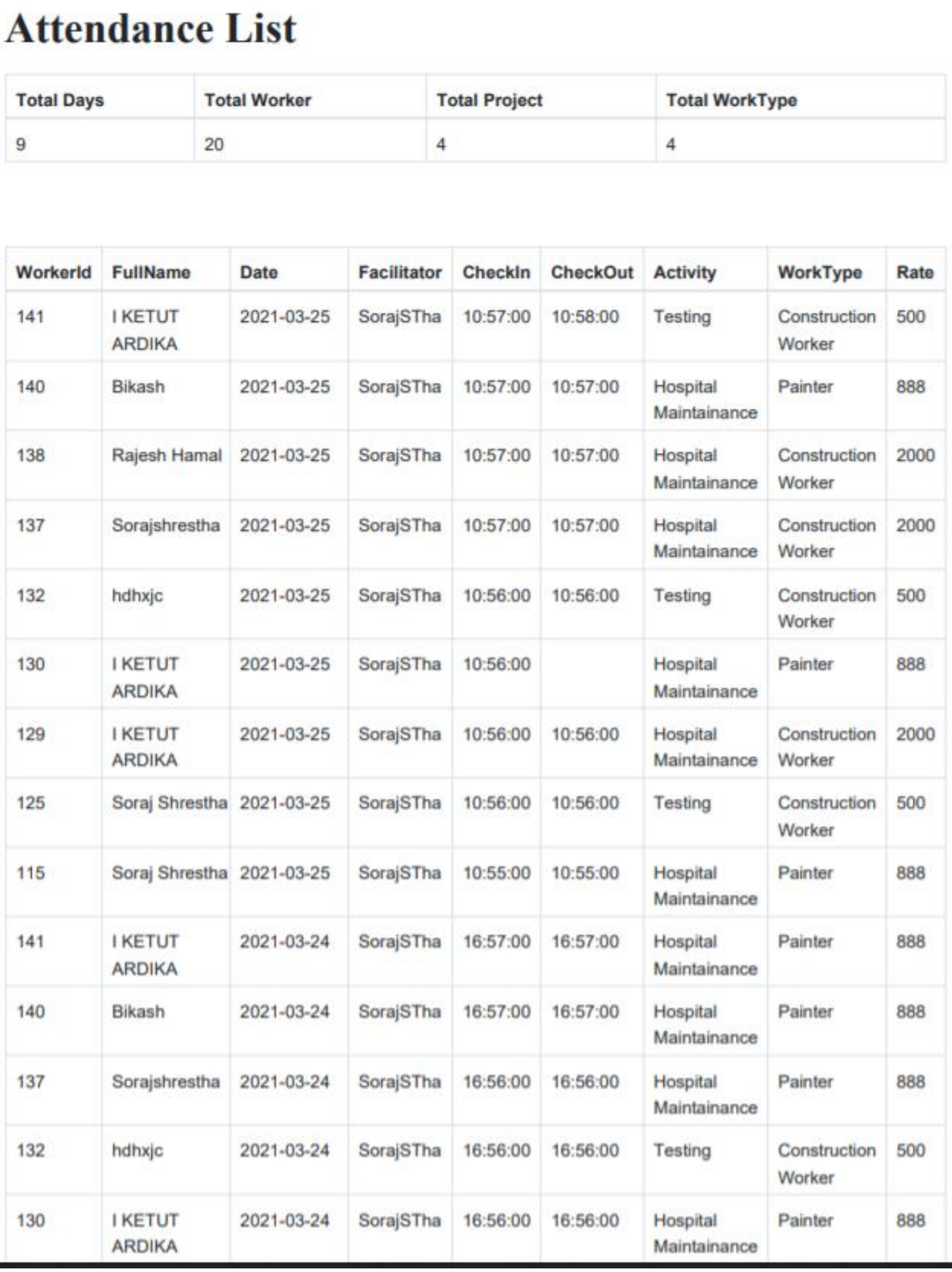
The system is also featured to make reports in various formats. We can modify our report data using the API that team has developed. Below are the details of the API for attendance report generation:

End Point: (baseurl) /CashforWorkLaravel/api/generate-pdf
Method: GET
Request Body: Authorization (Bearer Token of facilitator)
Parameters: as shown in img below:



Name	Description
Authorization * required string *(header)*	Bearer token of facilitator Authorization - Bearer token of facilitator
workerScanId string *(query)*	Worker Scan Id of worker workerScanId - Worker Scan Id of worker
name string *(query)*	Name of worker name - Name of worker
activityId number *(query)*	Activity Id for project activityId - Activity Id for project
workTypeId number *(query)*	Work Type for project workTypeId - Work Type for project
gender string *(query)*	Gender of worker gender - Gender of worker
fromDate string *(query)*	Date from where attendance is needed in yyyy-mm-dd format fromDate - Date from where attendance is ne
toDate string *(query)*	Date up to where attendance is needed in yyyy-mm-dd format toDate - Date up to where attendance is neec
date string *(query)*	Specific date for attendance in yyyy-mm-dd format date - Specific date for attendance in yyyy-mı

To create a report, we can click on the download button and select the format for which we want to export our data. An example of report generated by the system can be viewed below:



Attendance List

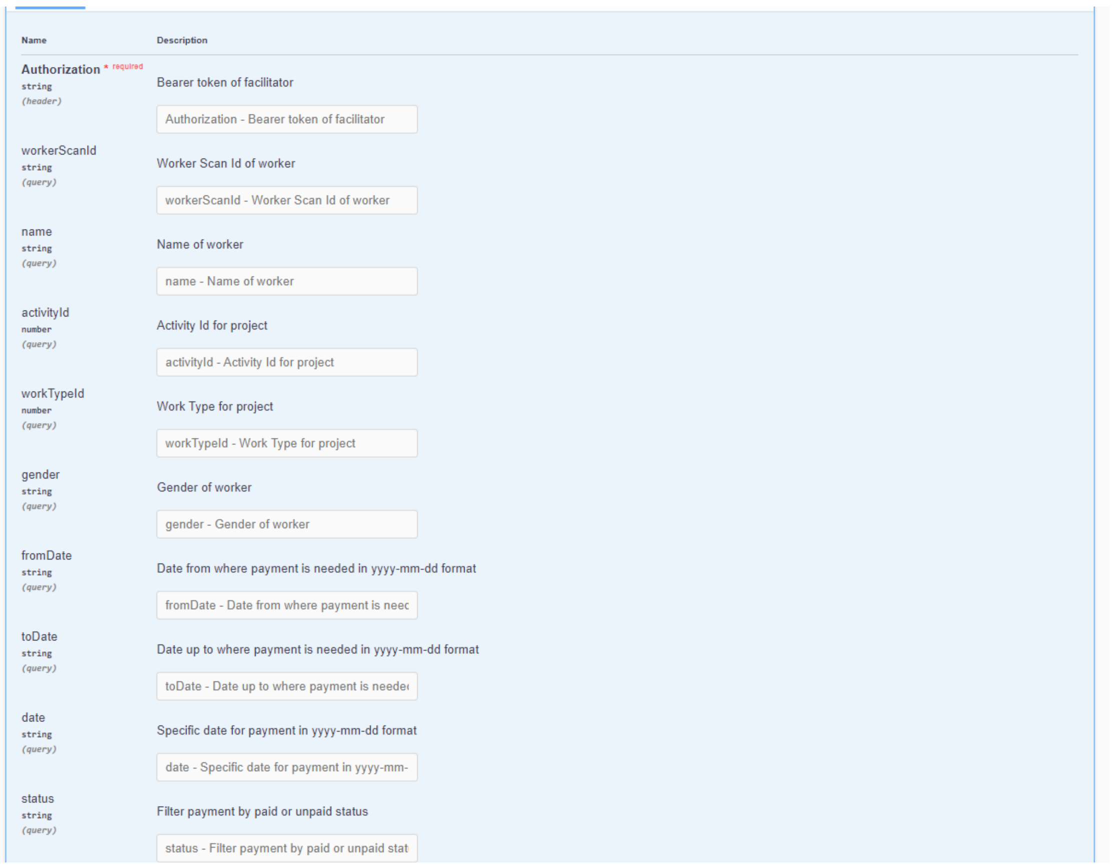
Total Days	Total Worker	Total Project	Total WorkType
9	20	4	4

WorkerId	FullName	Date	Facilitator	CheckIn	CheckOut	Activity	WorkType	Rate
141	I KETUT ARDIKA	2021-03-25	SorajSTha	10:57:00	10:58:00	Testing	Construction Worker	500
140	Bikash	2021-03-25	SorajSTha	10:57:00	10:57:00	Hospital Maintainance	Painter	888
138	Rajesh Hamal	2021-03-25	SorajSTha	10:57:00	10:57:00	Hospital Maintainance	Construction Worker	2000
137	Sorajshrestha	2021-03-25	SorajSTha	10:57:00	10:57:00	Hospital Maintainance	Construction Worker	2000
132	hdhxjc	2021-03-25	SorajSTha	10:56:00	10:56:00	Testing	Construction Worker	500
130	I KETUT ARDIKA	2021-03-25	SorajSTha	10:56:00		Hospital Maintainance	Painter	888
129	I KETUT ARDIKA	2021-03-25	SorajSTha	10:56:00	10:56:00	Hospital Maintainance	Construction Worker	2000
125	Soraj Shrestha	2021-03-25	SorajSTha	10:56:00	10:56:00	Testing	Construction Worker	500
115	Soraj Shrestha	2021-03-25	SorajSTha	10:55:00	10:55:00	Hospital Maintainance	Painter	888
141	I KETUT ARDIKA	2021-03-24	SorajSTha	16:57:00	16:57:00	Hospital Maintainance	Painter	888
140	Bikash	2021-03-24	SorajSTha	16:57:00	16:57:00	Hospital Maintainance	Painter	888
137	Sorajshrestha	2021-03-24	SorajSTha	16:56:00	16:56:00	Hospital Maintainance	Painter	888
132	hdhxjc	2021-03-24	SorajSTha	16:56:00	16:56:00	Testing	Construction Worker	500
130	I KETUT ARDIKA	2021-03-24	SorajSTha	16:56:00	16:56:00	Hospital Maintainance	Painter	888

PAYMENT MODULE

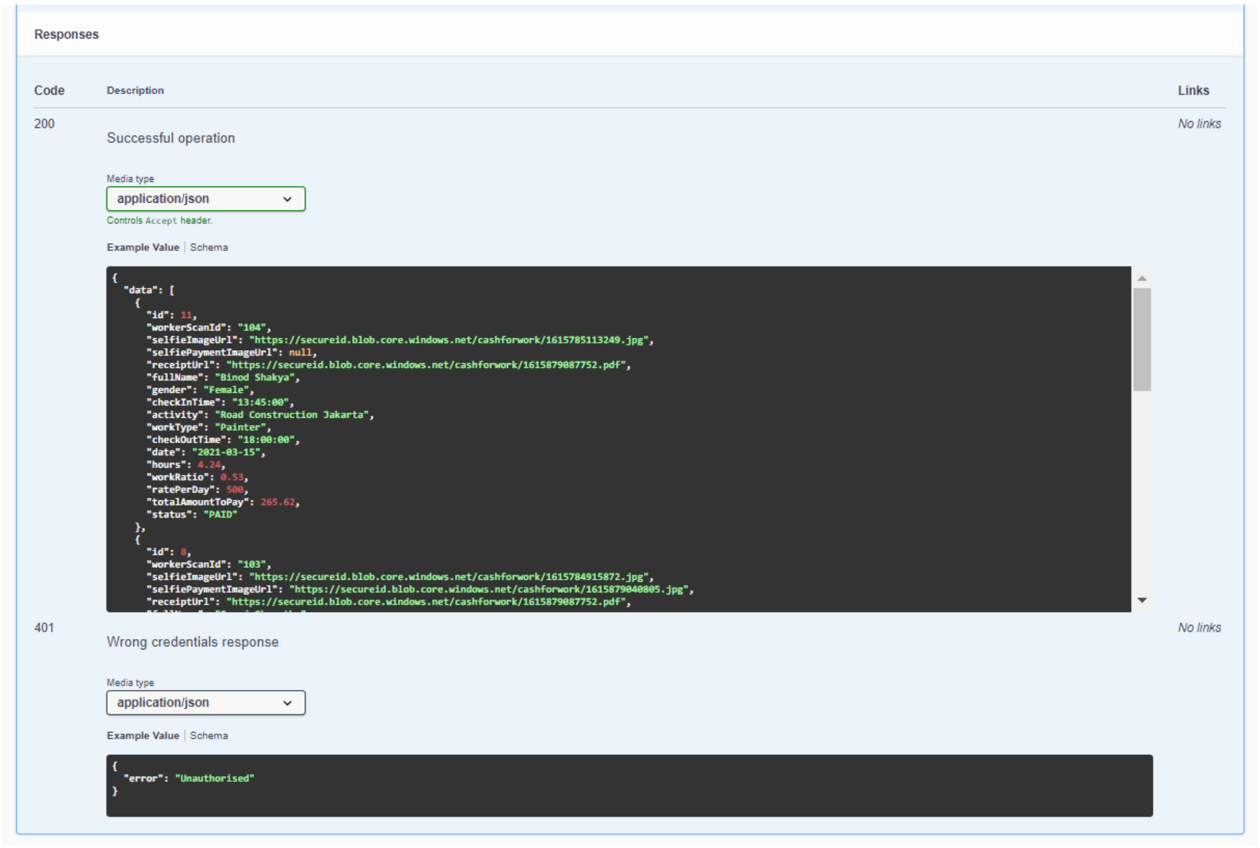
API FOR FETCHING PAYMENT LIST

To get list of payments on application and webportal, this API manages the request to fetch the payments with required filters required. The API is designed in such a way that when user requests for payment list by passing certain filters, it returns result based on it. For example if user wants payment of a particular date or of a certain gender or maybe any other filter this API collects the requirement and sends the payment list accordingly. Details are as follows:
End Point: (baseurl) /CashforWorkLaravel/api/payment
Method: GET
Request Body: Authorization (Bearer Token of facilitator)
Parameters: (shown in pic below)



Name	Description
Authorization * required string (header)	Bearer token of facilitator Authorization - Bearer token of facilitator
workerScanId string (query)	Worker Scan Id of worker workerScanId - Worker Scan Id of worker
name string (query)	Name of worker name - Name of worker
activityId number (query)	Activity Id for project activityId - Activity Id for project
workTypeId number (query)	Work Type for project workTypeId - Work Type for project
gender string (query)	Gender of worker gender - Gender of worker
fromDate string (query)	Date from where payment is needed in yyyy-mm-dd format fromDate - Date from where payment is neec
toDate string (query)	Date up to where payment is needed in yyyy-mm-dd format toDate - Date up to where payment is needed
date string (query)	Specific date for payment in yyyy-mm-dd format date - Specific date for payment in yyyy-mm-
status string (query)	Filter payment by paid or unpaid status status - Filter payment by paid or unpaid stat

If any parameter is not sent then it sends list of all the payment. But usually when facilitator requests for his/her workers payment, token is passed and only payment list of their worker is sent. The response received is as shown below:



Responses

Code	Description	Links
200	Successful operation	No links

Media type: application/json
Controls Accept header.

Example Value | Schema

```
{
  "data": [
    {
      "id": 11,
      "workerScanId": "104",
      "selfieImageUrl": "https://secureid.blob.core.windows.net/cashforwork/1615785113249.jpg",
      "selfiePaymentImageUrl": null,
      "receiptUrl": "https://secureid.blob.core.windows.net/cashforwork/1615879087752.pdf",
      "fullName": "Binod Shakya",
      "gender": "Female",
      "checkInTime": "13:45:00",
      "activity": "Road Construction Jakarta",
      "workType": "Painter",
      "checkOutTime": "18:00:00",
      "date": "2021-03-15",
      "hours": 4.24,
      "workRatio": 0.53,
      "ratePerDay": 500,
      "totalAmountToPay": 265.62,
      "status": "PAID"
    },
    {
      "id": 8,
      "workerScanId": "103",
      "selfieImageUrl": "https://secureid.blob.core.windows.net/cashforwork/1615784915872.jpg",
      "selfiePaymentImageUrl": "https://secureid.blob.core.windows.net/cashforwork/1615879040805.jpg",
      "receiptUrl": "https://secureid.blob.core.windows.net/cashforwork/1615879087752.pdf",
```

Code	Description	Links
401	Wrong credentials response	No links

Media type: application/json

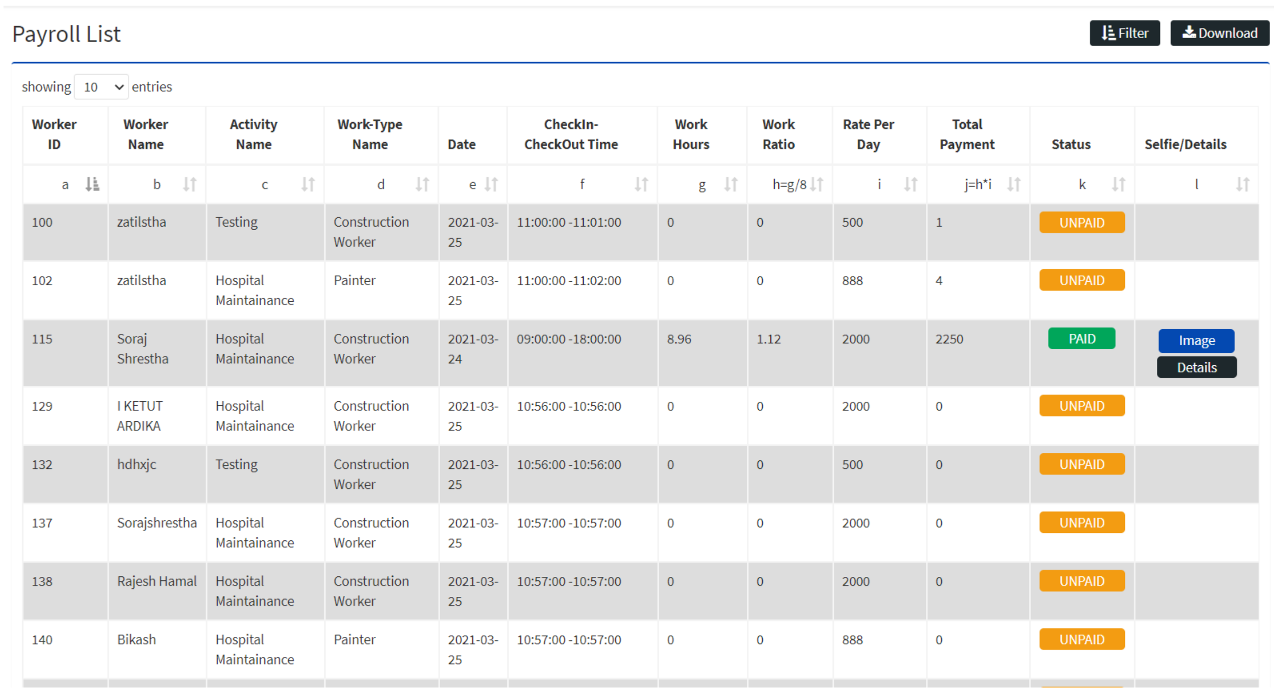
Example Value | Schema

```
{
  "error": "Unauthorised"
}
```

VIEWS FOR PAYMENT LIST

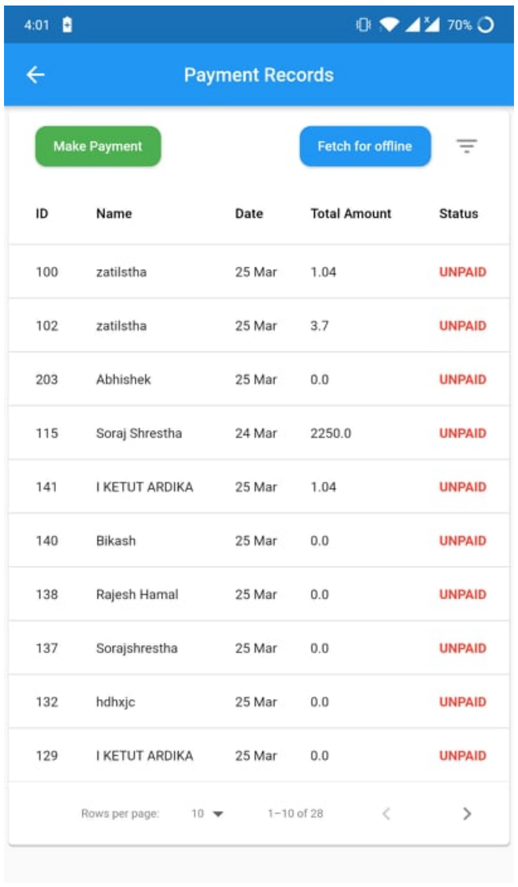
The payment module is designed in a way that displays payment from recent date to earlier ones. Each facilitator will be displayed payments of their concerned workers. If no any filter value is sent all the payments can be viewed. If we want payment detail in a specific form then we can use the filter option given in both app and web portal.

The screenshot below demonstrates the different views of the payment model


Payroll List

showing 10 entries

Worker ID	Worker Name	Activity Name	Work-Type Name	Date	CheckIn-CheckOut Time	Work Hours	Work Ratio	Rate Per Day	Total Payment	Status	Selfie/Details
a	b	c	d	e	f	g	h=g/8	i	j=h*i	k	l
100	zatilstha	Testing	Construction Worker	2021-03-25	11:00:00 -11:01:00	0	0	500	1	UNPAID	
102	zatilstha	Hospital Maintainance	Painter	2021-03-25	11:00:00 -11:02:00	0	0	888	4	UNPAID	
115	Soraj Shrestha	Hospital Maintainance	Construction Worker	2021-03-24	09:00:00 -18:00:00	8.96	1.12	2000	2250	PAID	Image Details
129	I KETUT ARDIKA	Hospital Maintainance	Construction Worker	2021-03-25	10:56:00 -10:56:00	0	0	2000	0	UNPAID	
132	hdhxjc	Testing	Construction Worker	2021-03-25	10:56:00 -10:56:00	0	0	500	0	UNPAID	
137	Sorajshrestha	Hospital Maintainance	Construction Worker	2021-03-25	10:57:00 -10:57:00	0	0	2000	0	UNPAID	
138	Rajesh Hamal	Hospital Maintainance	Construction Worker	2021-03-25	10:57:00 -10:57:00	0	0	2000	0	UNPAID	
140	Bikash	Hospital Maintainance	Painter	2021-03-25	10:57:00 -10:57:00	0	0	888	0	UNPAID	

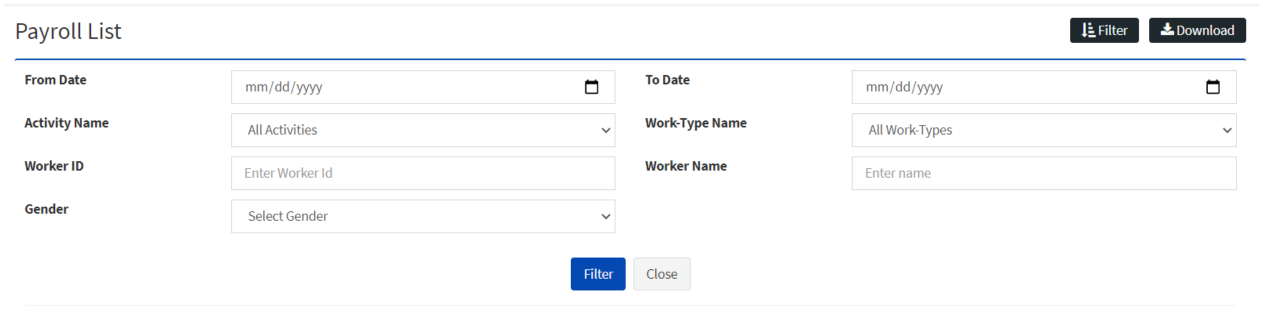

Payment Records

ID	Name	Date	Total Amount	Status
100	zatilstha	25 Mar	1.04	UNPAID
102	zatilstha	25 Mar	3.7	UNPAID
203	Abhishek	25 Mar	0.0	UNPAID
115	Soraj Shrestha	24 Mar	2250.0	UNPAID
141	I KETUT ARDIKA	25 Mar	1.04	UNPAID
140	Bikash	25 Mar	0.0	UNPAID
138	Rajesh Hamal	25 Mar	0.0	UNPAID
137	Sorajshrestha	25 Mar	0.0	UNPAID
132	hdhxjc	25 Mar	0.0	UNPAID
129	I KETUT ARDIKA	25 Mar	0.0	UNPAID

Rows per page: 10 1–10 of 28

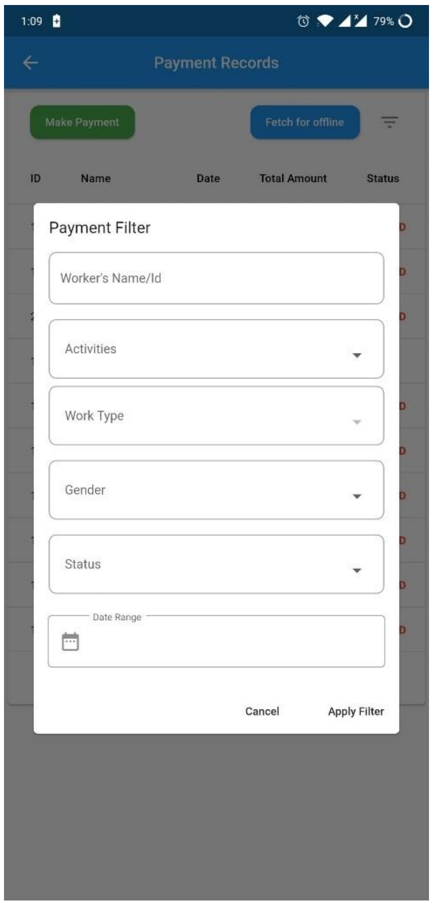
The two pictures are an example of what facilitator/admin could view when they click on the payroll tab. Initially, all of his/her worker`s payments are listed by recent date first order. On the app there is a Make Payment button as well to make payments to the worker. To filter out this payment list there is filter button at the top of the screen. There are various fields from which we can filter out the data which is shown in the sceenshot below:

FILTERING DATA



Payroll List

From Date	mm/dd/yyyy	To Date	mm/dd/yyyy
Activity Name	All Activities	Work-Type Name	All Work-Types
Worker ID	Enter Worker Id	Worker Name	Enter name
Gender	Select Gender		



As seen on the screenshot, we can filter the data using parameters such as:

* Payments of a single worker using his/her id or name. This list will be filtered out to display only selected persons data.

* Payments made on a particular activity. This will filter data to display list of payments on selected activity.

*Payments made on certain work group type

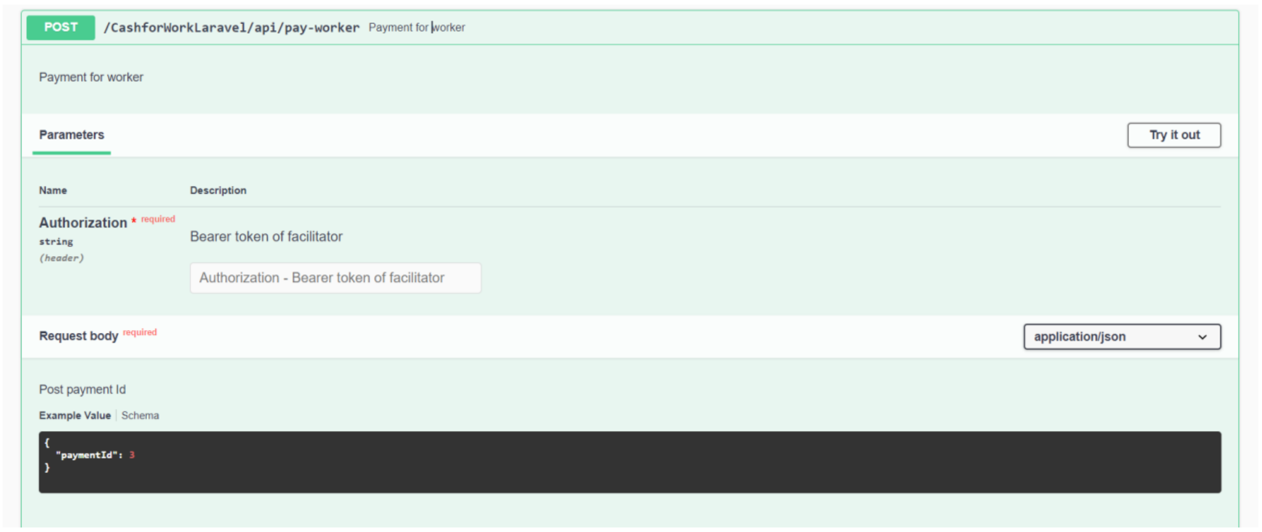
*Payment list of specific gender group

*Payments of different status. Paid and unda

*Payment of a range of days by selecting date.

MAKING PAYMENTS

The API used to make payments to the worker is shown below:
End Point: (baseurl) /CashforWorkLaravel/api/pay-worker
Method: POST
Request Body: Authorization (Bearer Token of facilitator)
Parameters: Payment ID,


POST /CashforWorkLaravel/api/pay-worker Payment for worker

Payment for worker

Parameters

Name	Description
Authorization * required string (header)	Bearer token of facilitator Authorization - Bearer token of facilitator

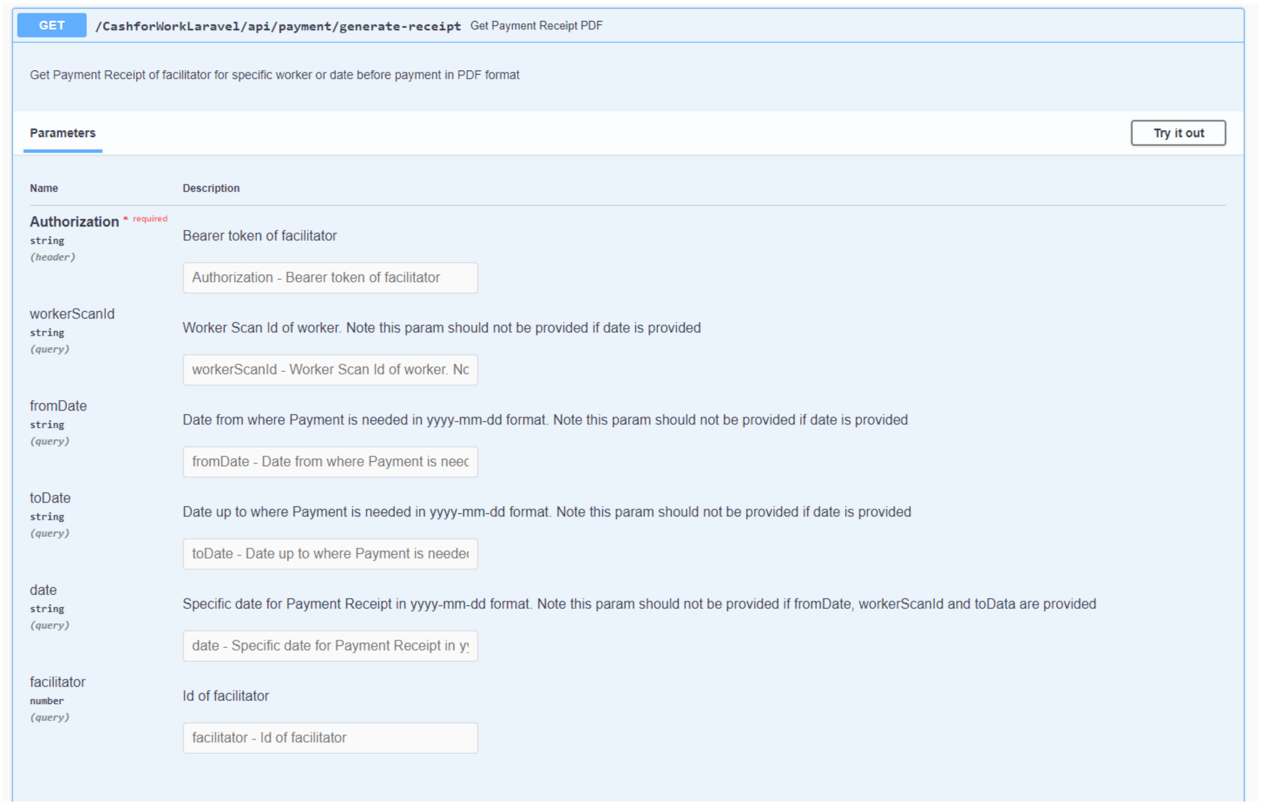
Request body required — application/json

Post payment Id

Example Value | Schema

```
{
  "paymentId": 3
}
```

According to the flow before payments can be made a receipt would have to be generated so that facilitators could take them to get signature from worker upon getting payment.
The API used to generate the receipt is shown below:
End Point: (baseurl) /CashforWorkLaravel/api/payment/generate-receipt
Method: GET
Request Body: Authorization (Bearer Token of facilitator)
Parameters: Worker ID, fromdate, todate, facilitator



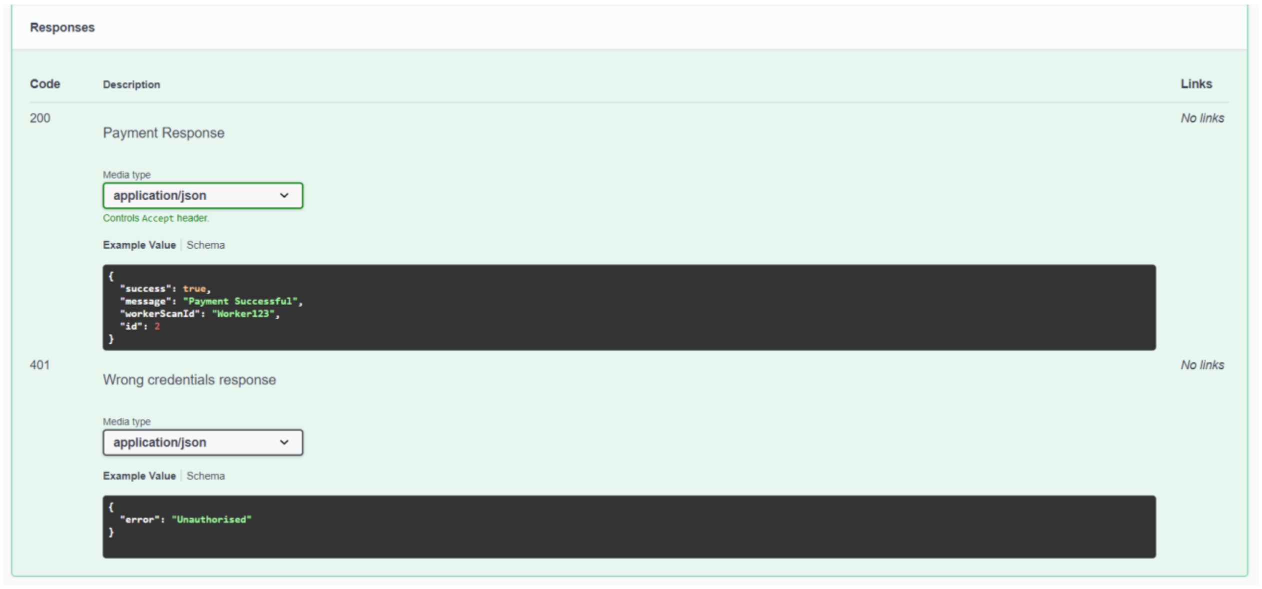
GET `/CashforWorkLaravel/api/payment/generate-receipt` Get Payment Receipt PDF

Get Payment Receipt of facilitator for specific worker or date before payment in PDF format

Parameters

Name	Description
Authorization * required `string` *(header)*	Bearer token of facilitator Authorization - Bearer token of facilitator
workerScanId `string` *(query)*	Worker Scan Id of worker. Note this param should not be provided if date is provided workerScanId - Worker Scan Id of worker. Nc
fromDate `string` *(query)*	Date from where Payment is needed in yyyy-mm-dd format. Note this param should not be provided if date is provided fromDate - Date from where Payment is neec
toDate `string` *(query)*	Date up to where Payment is needed in yyyy-mm-dd format. Note this param should not be provided if date is provided toDate - Date up to where Payment is neede
date `string` *(query)*	Specific date for Payment Receipt in yyyy-mm-dd format. Note this param should not be provided if fromDate, workerScanId and toData are provided date - Specific date for Payment Receipt in y
facilitator `number` *(query)*	Id of facilitator facilitator - Id of facilitator

The response we receive on successful API hit is shown below:



Responses

Code	Description	Links
200	Payment Response	*No links*
401	Wrong credentials response	*No links*

200 — Media type: application/json. Controls `Accept` header.

Example Value | Schema

```
{
  "success": true,
  "message": "Payment Successful",
  "workerScanId": "Worker123",
  "id": 2
}
```

401 — Media type: application/json

Example Value | Schema

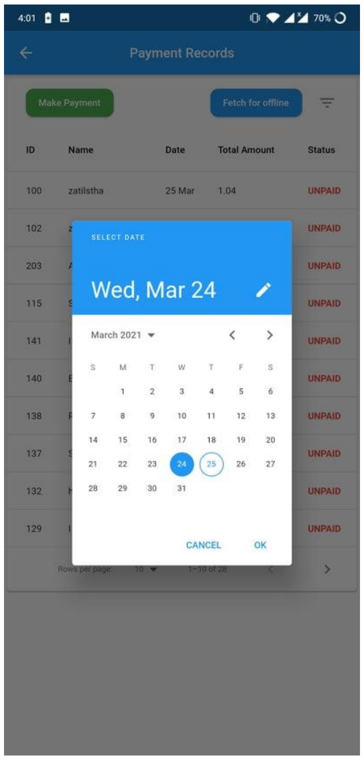
```
{
  "error": "Unauthorised"
}
```

UI TO MAKE PAYMENTS

To make the payment, there is a Make payment button on the top of the payment page. When we click on it, we are redirected to select a date for which we want to make payment. Then a list of payments for that day are listed out. Next, we find the worker for whom we want to make payment and then click the pay button to proceed further.

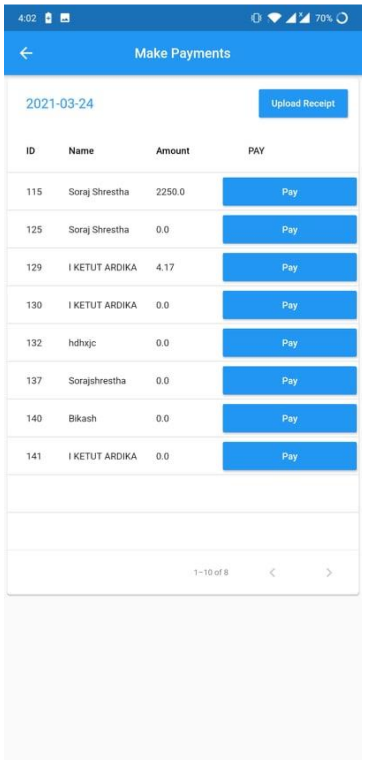
During online payment first a selfie is taken and verified against the initial image during registration. The face match result is displayed and payment process can be moved ahead. The result of the face match is also displayed in the web portal. A signature on the receipt is also taken as proof and for future reference after payment is made which can be uploaded manually from app as well as web portal.

Images shown are demonstration of the process to make payment when application is online:



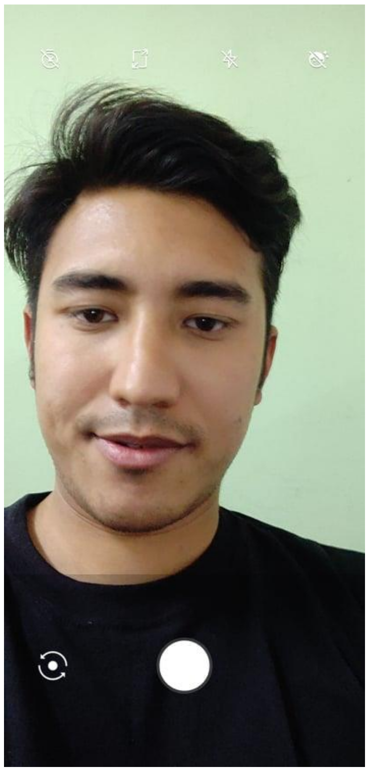
ID	Name	Date	Total Amount	Status
100	zatilstha	25 Mar	1.04	UNPAID
102	z			UNPAID
203	A			UNPAID
115	S			UNPAID
141	I			UNPAID
140	E			UNPAID
138	F			UNPAID
137	S			UNPAID
132	H			UNPAID
129	I			UNPAID

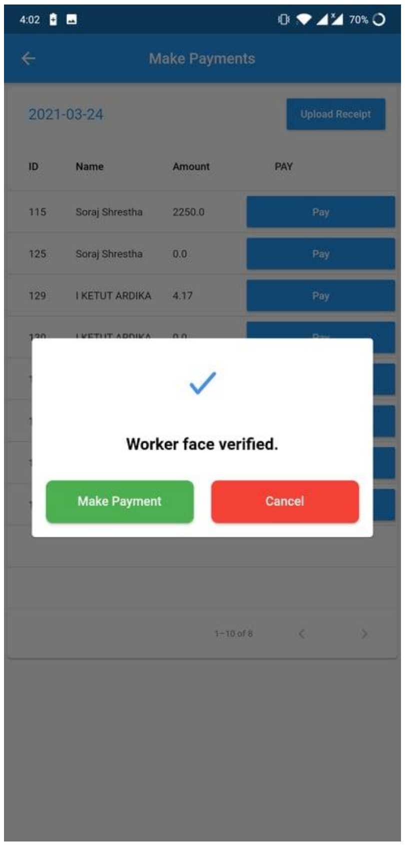
S	M	T	W	T	F	S
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31			



2021-03-24

ID	Name	Amount	PAY
115	Soraj Shrestha	2250.0	Pay
125	Soraj Shrestha	0.0	Pay
129	I KETUT ARDIKA	4.17	Pay
130	I KETUT ARDIKA	0.0	Pay
132	hdhxjc	0.0	Pay
137	Sorajshrestha	0.0	Pay
140	Bikash	0.0	Pay
141	I KETUT ARDIKA	0.0	Pay




4:02 · 70%

Make Payments

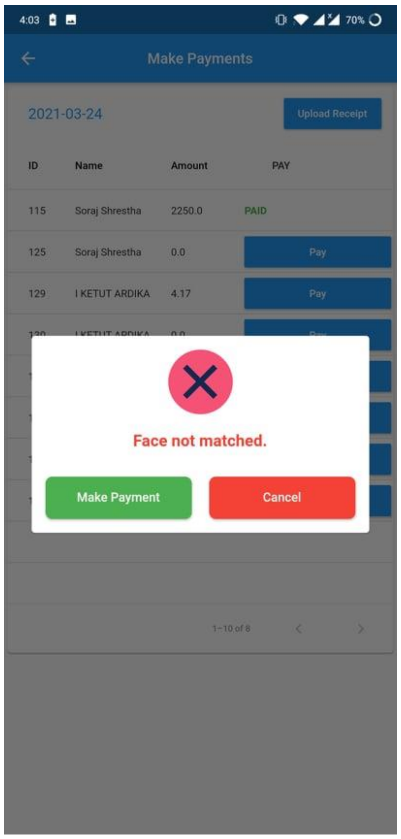
2021-03-24 · Upload Receipt

ID	Name	Amount	PAY
115	Soraj Shrestha	2250.0	Pay
125	Soraj Shrestha	0.0	Pay
129	I KETUT ARDIKA	4.17	Pay
130	I KETUT ARDIKA	0.0	Pay

Worker face verified.

Make Payment · Cancel

1–10 of 8


4:03 · 70%

Make Payments

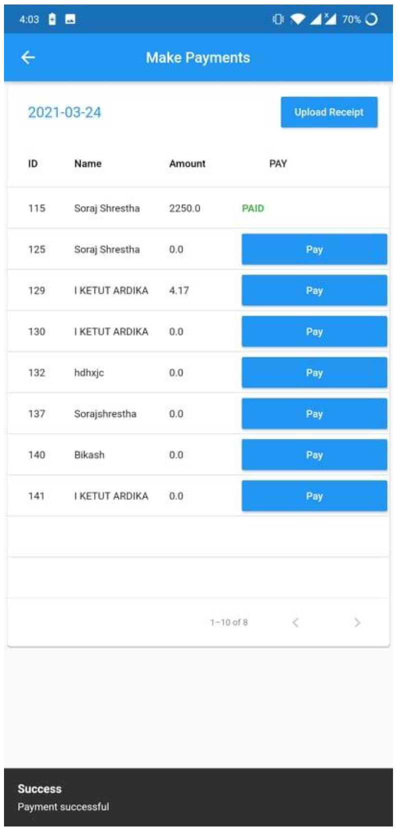
2021-03-24 · Upload Receipt

ID	Name	Amount	PAY
115	Soraj Shrestha	2250.0	PAID
125	Soraj Shrestha	0.0	Pay
129	I KETUT ARDIKA	4.17	Pay
130	I KETUT ARDIKA	0.0	Pay

Face not matched.

Make Payment · Cancel

1–10 of 8


4:03 · 70%

Make Payments

2021-03-24 · Upload Receipt

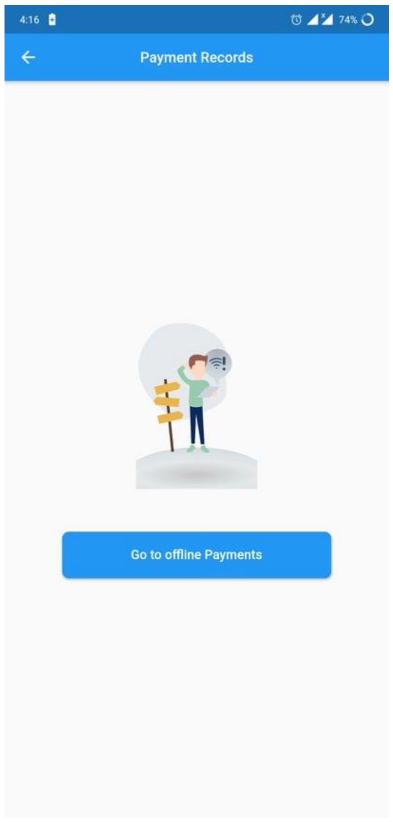
ID	Name	Amount	PAY
115	Soraj Shrestha	2250.0	PAID
125	Soraj Shrestha	0.0	Pay
129	I KETUT ARDIKA	4.17	Pay
130	I KETUT ARDIKA	0.0	Pay
132	hdhxjc	0.0	Pay
137	Sorajshrestha	0.0	Pay
140	Bikash	0.0	Pay
141	I KETUT ARDIKA	0.0	Pay

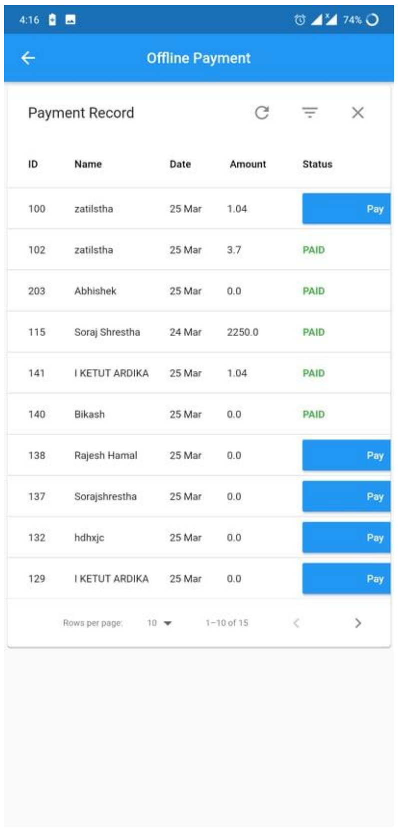
1–10 of 8

Success
Payment successful

OFFLINE PAYMENT SYSTEM

If the application is in online state then payment data is synced in to the system but if in case there is no internet connectivity then first, we will have to fetch payment data using fetch 1 week data with internet connectivity button. After then even if we are in offline state, we can make payment using the fetched data of past 1 week. No face verification is required for offline payment, but these data are stored in local database and we will have to sync the data to send it to the system. We can view our unsynced data by clicking the chart on the home screen. Screenshot below is an example of unsynced data on our device.

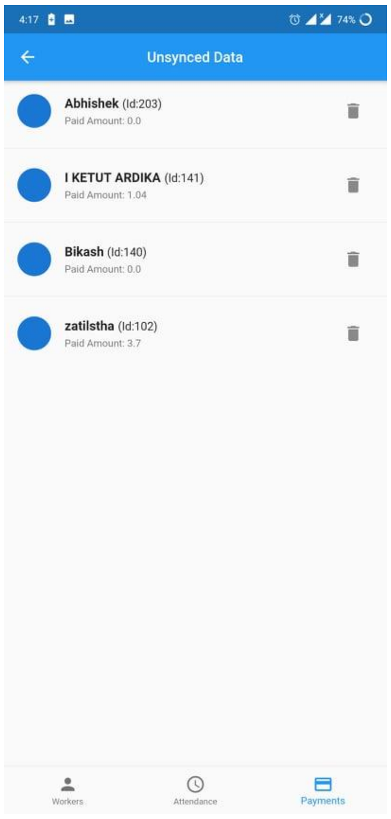




Payment Record

ID	Name	Date	Amount	Status
100	zatilstha	25 Mar	1.04	Pay
102	zatilstha	25 Mar	3.7	PAID
203	Abhishek	25 Mar	0.0	PAID
115	Soraj Shrestha	24 Mar	2250.0	PAID
141	I KETUT ARDIKA	25 Mar	1.04	PAID
140	Bikash	25 Mar	0.0	PAID
138	Rajesh Hamal	25 Mar	0.0	Pay
137	Sorajshrestha	25 Mar	0.0	Pay
132	hdhxjc	25 Mar	0.0	Pay
129	I KETUT ARDIKA	25 Mar	0.0	Pay

Rows per page: 10 1–10 of 15



Abhishek (Id:203)
Paid Amount: 0.0

I KETUT ARDIKA (Id:141)
Paid Amount: 1.04

Bikash (Id:140)
Paid Amount: 0.0

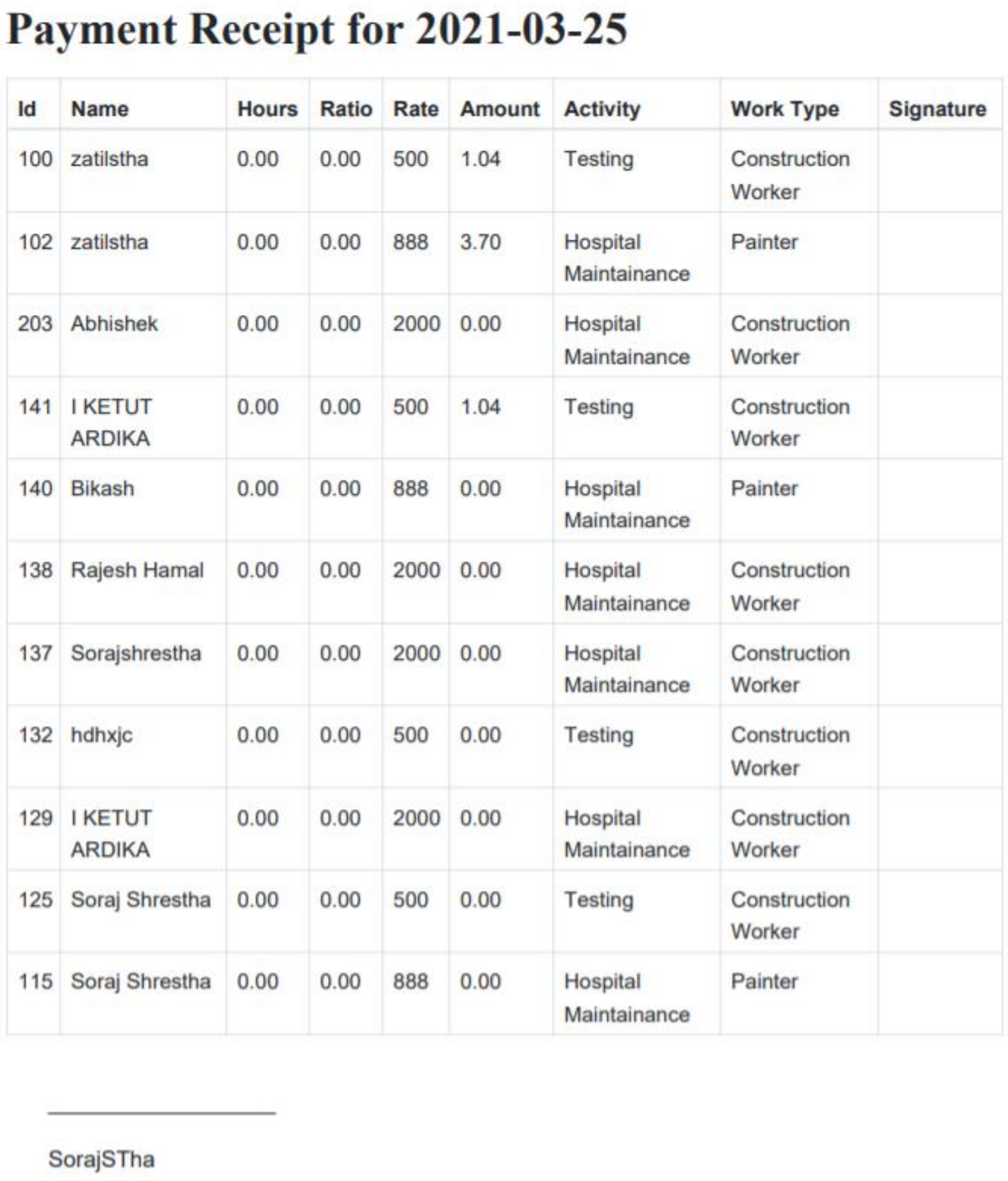
zatilstha (Id:102)
Paid Amount: 3.7

PAYMENT RECEIPT

The application also features a generating receipt section where facilitators will have to go to the portal and create a reciept for a day for which he wants to make the recipt. This reciept can now be printed and taken to the field. When he makes payment then he can collect a signature of the worker who is paid. This can be used as a proof for future and will also be useful for keeping records. The process to generate receipt are as follows:

- First go to payment receipt section.
- Select date and facilitaor.
- A link of the recipt will be generated after some time which can be printed out
- There is section to upload the signed reciept once the payment process is complete.

Image Below is an example of a payment receipt that is given by the system:



Payment Receipt for 2021-03-25

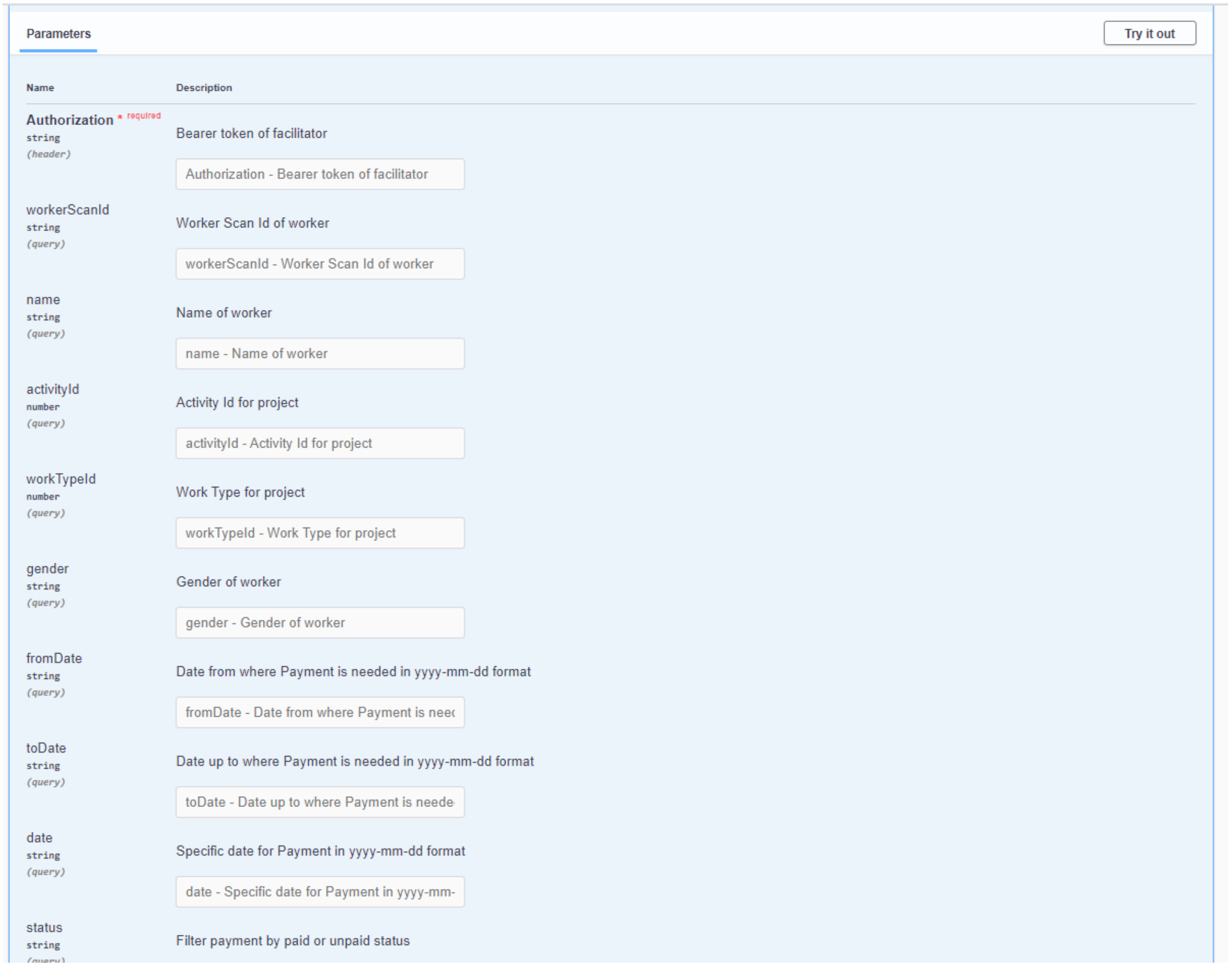
Id	Name	Hours	Ratio	Rate	Amount	Activity	Work Type	Signature
100	zatilstha	0.00	0.00	500	1.04	Testing	Construction Worker	
102	zatilstha	0.00	0.00	888	3.70	Hospital Maintainance	Painter	
203	Abhishek	0.00	0.00	2000	0.00	Hospital Maintainance	Construction Worker	
141	I KETUT ARDIKA	0.00	0.00	500	1.04	Testing	Construction Worker	
140	Bikash	0.00	0.00	888	0.00	Hospital Maintainance	Painter	
138	Rajesh Hamal	0.00	0.00	2000	0.00	Hospital Maintainance	Construction Worker	
137	Sorajshrestha	0.00	0.00	2000	0.00	Hospital Maintainance	Construction Worker	
132	hdhxjc	0.00	0.00	500	0.00	Testing	Construction Worker	
129	I KETUT ARDIKA	0.00	0.00	2000	0.00	Hospital Maintainance	Construction Worker	
125	Soraj Shrestha	0.00	0.00	500	0.00	Testing	Construction Worker	
115	Soraj Shrestha	0.00	0.00	888	0.00	Hospital Maintainance	Painter	

SorajSTha

GENERATING PAYMENT REPORT

The system is also featured to make reports in various formats. We can modify our report data using the API that team has developed. Below are the details of the API for payment report generation:

End Point: (baseurl) /CashforWorkLaravel/api/payment/generate-pdf
Method: GET
Request Body: Authorization (Bearer Token of facilitator)
Parameters: as shown in img below:

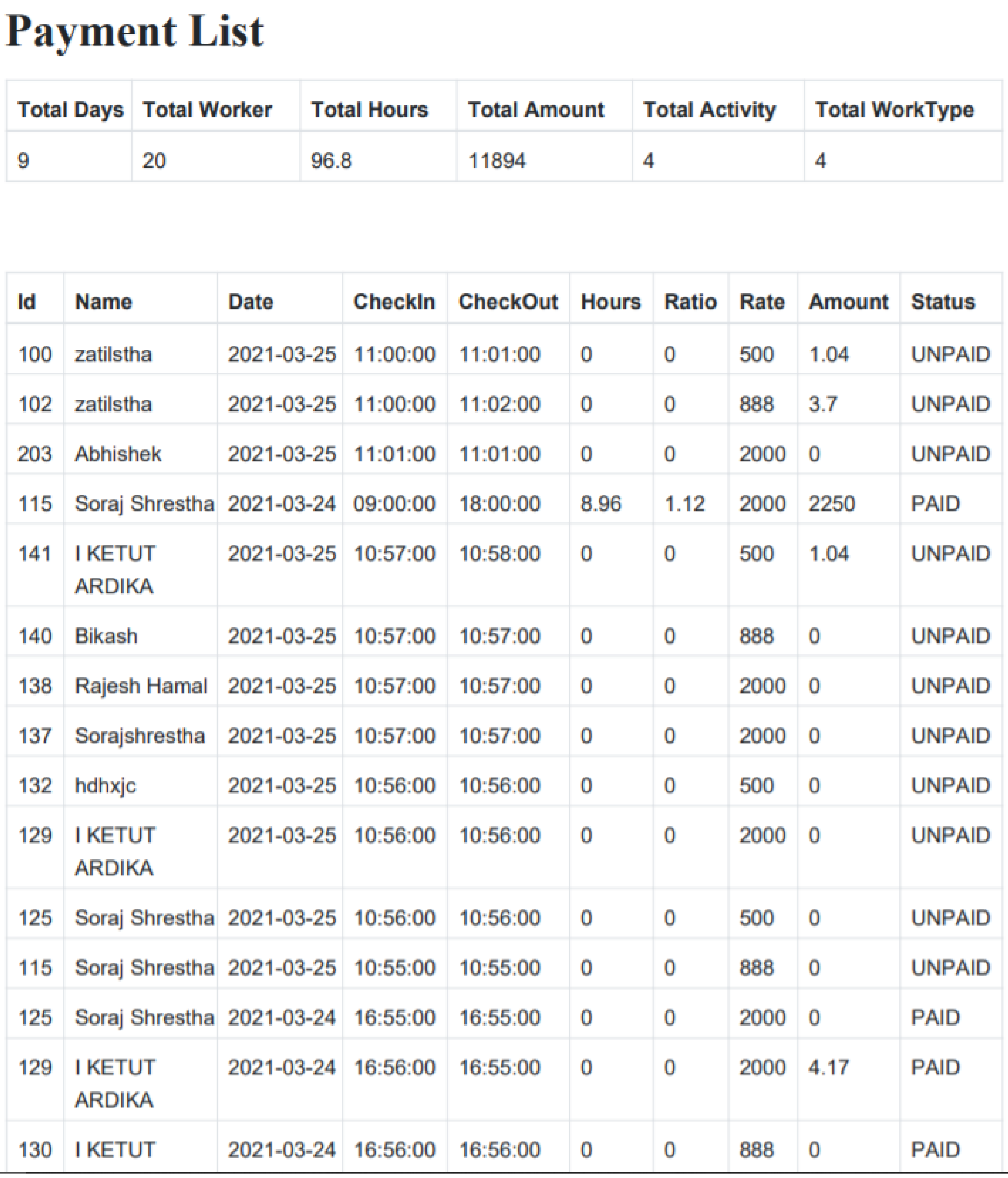


Parameters

Try it out

Name	Description
Authorization * required string *(header)*	Bearer token of facilitator Authorization - Bearer token of facilitator
workerScanId string *(query)*	Worker Scan Id of worker workerScanId - Worker Scan Id of worker
name string *(query)*	Name of worker name - Name of worker
activityId number *(query)*	Activity Id for project activityId - Activity Id for project
workTypeId number *(query)*	Work Type for project workTypeId - Work Type for project
gender string *(query)*	Gender of worker gender - Gender of worker
fromDate string *(query)*	Date from where Payment is needed in yyyy-mm-dd format fromDate - Date from where Payment is neec
toDate string *(query)*	Date up to where Payment is needed in yyyy-mm-dd format toDate - Date up to where Payment is neede
date string *(query)*	Specific date for Payment in yyyy-mm-dd format date - Specific date for Payment in yyyy-mm-
status string	Filter payment by paid or unpaid status

To create a report, we can click on the download button and select the format for which we want to export our data. An example of report generated by the system can be viewed below:



Payment List

Total Days	Total Worker	Total Hours	Total Amount	Total Activity	Total WorkType
9	20	96.8	11894	4	4

Id	Name	Date	CheckIn	CheckOut	Hours	Ratio	Rate	Amount	Status
100	zatilstha	2021-03-25	11:00:00	11:01:00	0	0	500	1.04	UNPAID
102	zatilstha	2021-03-25	11:00:00	11:02:00	0	0	888	3.7	UNPAID
203	Abhishek	2021-03-25	11:01:00	11:01:00	0	0	2000	0	UNPAID
115	Soraj Shrestha	2021-03-24	09:00:00	18:00:00	8.96	1.12	2000	2250	PAID
141	I KETUT ARDIKA	2021-03-25	10:57:00	10:58:00	0	0	500	1.04	UNPAID
140	Bikash	2021-03-25	10:57:00	10:57:00	0	0	888	0	UNPAID
138	Rajesh Hamal	2021-03-25	10:57:00	10:57:00	0	0	2000	0	UNPAID
137	Sorajshrestha	2021-03-25	10:57:00	10:57:00	0	0	2000	0	UNPAID
132	hdhxjc	2021-03-25	10:56:00	10:56:00	0	0	500	0	UNPAID
129	I KETUT ARDIKA	2021-03-25	10:56:00	10:56:00	0	0	2000	0	UNPAID
125	Soraj Shrestha	2021-03-25	10:56:00	10:56:00	0	0	500	0	UNPAID
115	Soraj Shrestha	2021-03-25	10:55:00	10:55:00	0	0	888	0	UNPAID
125	Soraj Shrestha	2021-03-24	16:55:00	16:55:00	0	0	2000	0	PAID
129	I KETUT ARDIKA	2021-03-24	16:56:00	16:55:00	0	0	2000	4.17	PAID
130	I KETUT	2021-03-24	16:56:00	16:56:00	0	0	888	0	PAID

CONCLUSION

To sum up the entire document presented above, it can be said that the development phase of the solution for cash for work has been completed. All the necessary modules have been developed with proper UI. Based on weekly meetings few initial requirements have been changed but the solution is now capable of meeting all the requirements.

The next step would be to make the solution bug free for which a series of testing are being carried out. Planning to take the solution for test run and pilot run of the solution has also been thought of. This should ensure proper and smooth operation of solution during actual implementation in the field.