iSchoolOle monthly plan test link: <https://paystack.shop/pay/q0kthomq0j>

ID: PLN\_xxxxMonthly

iSchoolOle yearly plan test link: <https://paystack.shop/pay/td8as0suef>

ID: PLN\_xxxxYearly

**New Ole Email**

DOB: June 11th, 1985

Email: [ischoolole@gmail.com](mailto:ischoolole@gmail.com)

Password: @IschooOle24

Paystack (Personal)

Email: [ischoolole@gmail.com](mailto:ischoolole@gmail.com)

Password: @IschooOle24

[ischoolola@gmail.com](mailto:ischoolola@gmail.com)

Date of Birth: May, 21st 1985

Password: @iSchoolApp81

Paystack Login Details:

Email: [ischoolola@gmail.com](mailto:ischoolola@gmail.com)

Password: @iSchoolApp81

Phone Number: 09063883848

app/

├── \_layout.tsx <-- Global layout (used only for routing)

├── index.tsx <-- Splash screen (initial screen)

├── (drawer)/ <-- Screens that should have drawer

│ ├── \_layout.tsx <-- Drawer layout

│ ├── home.tsx <-- Home screen with drawer

│ ├── about-us.tsx <-- Example drawer screen

SENDGRID API KEY: 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

Would you like the splash screen to have a loading spinner animation as well?

Use this in production: email = f[{username}@ischool.ng](mailto:%7busername%7d@ischool.ng) In email settings.

<View style={styles.imageOverlay} />

app/

├── \_layout.tsx <-- Stack (root layout)

├── (tabs)/ <-- Tab screens live here

│ └── \_layout.tsx <-- TabLayout.tsx (bottom tabs)

│ └── index.tsx <-- Home Screen

│ └── explore.tsx <-- Explore tab

├── index.tsx <-- Splash screen

├── AboutUs.tsx <-- About us page  
├── SignUp.tsx <-- Sign up screen

├── StudentLogin.tsx <-- About us page

├── ContactUs.tsx <-- Contact us page

app/

├── (drawer)/

|. ├──\_layout.tsx 🡨 Contains drawer navigator

│ ├── about-us.tsx

│ ├── contact-us.tsx

│ ├── signup.tsx

│ ├── student-login.tsx  
 |. ├── home.tsx

| ├── student-dashborad.tsx

| ├── take-test.tsx

│ └── (tabs)/  
 | └── \_layout.tsx <-- TabLayout.tsx (bottom tabs)

│ └── index.tsx <-- Home Screen

│ └── explore.tsx <-- Explore tab

├── \_layout.tsx ← RootLayout

├── index.tsx. SplashScreen (excluded from drawer)

Offcanvas menu:

Colour: gradient of sky blue colour(top) and dark colour(down)

At the top of the offcanvas: the brand logo

Menus items

Home (link to the home screen)

About Us (link to about us Screen)

Buy slots & Register

Students Login

ContactUs

Follow Us:  
(Facebook, Instagram and Twitter Icons)

React Native App:

Since you’ve already built a React frontend and a Django + DRF backend, we’ll **extract shared logic**, restructure your codebase where necessary, and set up the mobile app using **best practices**

## 🚀 Phase 1: Project Planning & Setup

### ✅ Step 1: Define MVP Scope for the Mobile App

Your MVP should cover:

1. **Onboarding**
2. **Authentication (optional at MVP)**
3. **Home Screen**
4. Sign Up page (For users to Register and pay for slots, already done in React/Django)
5. Student Login Page (registered students can login using the auto generated credentials (ready in react/Django)
6. Student dashboard ( After student login: ready in React/Django)
7. Testpage (for students to take test: ready in React)
8. Class Level Selection (JSS1–SSS3)
9. Subject & Topic Selection (via DRF)
10. Test Flow (10-minute timer, one question at a time)
11. Answer Submission (MCQ & theory)
12. Result Display (with score and detailed feedback)

…Am only seeing the Expo Router’s default starter screen

app/

├── \_layout.tsx

└── tabs/

├── \_layout.tsx

└── index.tsx   
  
 └── home.tsx   
  
 └── explore.tsx

- npm run android

- npm run ios

- npm run web

logging in with one student's credentials but seeing another student's dashboard data

Saved Twilio /SendGrid code: 6Q9C99LER85AGCY7LD24LMMG

Would you like me to help add validation to ensure slots\_applied is always a positive number?

send\_mail(

subject="Gmail SMTP Test",

message="This is a test email sent via Gmail SMTP in Django.",

from\_email=None, # Uses DEFAULT\_FROM\_EMAIL

recipient\_list=["gintason@gmail.com"], # use your real address

fail\_silently=False,

)

Remote job qualities:

1. Effective communication
2. Self-confidence.

Tools Needed:

Visual CV: Website to revamp my cv

Enhanced CV.

Resume worded

Physical tools:

1. Internet Access (100mph per hour)
2. A GOOD MICROPHONE AND EAR PIECE (With noise cancellation)
3. Dollar Account

Create a portfolio for submission. (canva app to create a porfolio)

Create a cover letter that fits each job roles. (use chatgpt or openai)

JOB SITES FOR GET REMOTE JOBS:

1. Freelancer
2. Upwork
3. Linkedin
4. turing

🔥 Your FlutterwaveCheckout callback **does not call your Django /api/verify-and-register/ endpoint after successful payment**, so the registration is **never completed**, and no student accounts are created.

### ✅ Here's exactly what you should do:

You must **modify your handleFlutterwavePayment** function in your React SignUpPage.jsx so that when the payment is successful, it:

1. Calls the Django /api/verify-and-register/ endpoint.
2. Passes the relevant metadata and transaction data (transaction\_id, tx\_ref, etc).
3. Displays feedback to the user.

Would you like me to now generate the full working code for SignupPage.jsx based on the roles and logic we discussed (including dynamic fields for each account type)?

iSchool App

Features of the iSchool ola App.

This is an online assessment platform where students can take online tests. Like a CBT platform. It's for Secondary schools, to cover JSS 1 to SSS 3 classes

1. User Roles & Access

o There should be Admin, Teachers, Students, and Parents users roles

o There should be Role-specific permissions and features

2. User roles and access flow (Front end):

Would you like help previewing the PDF output locally using Django?

Page routes with React Router: import { BrowserRouter as Router, Routes, Route } from 'react-router-dom';

You can further enhance the UI by customizing Bootstrap themes or integrating additional libraries like **React Icons** for visual cues.

Let me know if you also want to display the profile\_photo, registration\_group, or last\_login from the backend response.

building the student registration step after successful payment?

[ischoolola@gmail.com](mailto:ischoolola@gmail.com)

Date of Birth: May, 21st 1985

Password: @iSchoolApp81

Paystack Login Details:

Email: [ischoolola@gmail.com](mailto:ischoolola@gmail.com)

Password: @iSchoolApp81

Phone Number: 09063883848

End user: Launches app, User sees Sign up page.

On the sign up page, users Select: Home, School or Accredited Referral in a drop down box.

After selecting a Home, school or Accredited Referral, each option should further Indicate:

1. State
2. Number of slots applied for

But If Accredited Referral is selected

Indicate:

Let me know if you'd like:

* Logs in your Django view to verify backend calls.
* Email confirmation to be sent after registration.
* Loader spinner instead of text.

1. State
2. Number of slots applied for
3. Account details (for commissions)

Requirements for Fluterewave Registration

www.flutterwave.com/login

Email: onyemaechinwa9@gmail.com

Password: @iSchoolApp81

Backend:

* Confirm payment (Payment confirmation before user creation)
* Automatically generate/ assign individual usernames/serial numbers
* Send username to end user via email.
* Username format: 001/S001/000 or 001/H001/000
* State/School/student or State/Home/student
* integrates **GPT-based theory grading**

End User:-

* Receive approved number of paid slots in their email
* Enter names/email address for each slot
* Automatically generate/ assign individual usernames/serial numbers
* (Only last 3 digits of School/Home username will change to reflect individual student serial numbers)
* The assigned student username generated automatically becomes their individual
* app login password

✅ Example Email Output

Subject: Welcome to iSchool Ola – Your Student Accounts

Dear Royal School,

Your registration was successful! Below are the student login details:

Email: student1\_45@ischoolola.com | Password: J8mZpw35

Email: student2\_45@ischoolola.com | Password: bYt9Zm45

Email: student3\_45@ischoolola.com | Password: Y6qPa23t

Visit https://ischoolola.com/login to get started.

SC2324265887251

✅ Done! The backend will now:

* Verify Flutterwave payment
* Save the transaction
* Assign paid slots
* Generate usernames
* Send email to the user

Let me know if you'd like toast notifications instead of alert() or want to convert this to use **React Hook Form + Yup** for more scalable validation.

* 4o

Now, we’ll proceed to build the **React frontend for the student login form**. This form will allow a student to:

* Enter their assigned email and password (as received via email).
* Authenticate using your Django backend.
* On success, be redirected to the student\_dash.

### ✅ What we've achieved so far:

1. **User gets auto-created after Flutterwave payment** (verify\_and\_register view ✅).
2. **Login credentials are emailed to the provided address** ✅.

**Bootstrap 5** for responsiveness and aesthetics

Student interface:

* Login
* Student name
* Assigned username
* Password
* Take a test.
* Maximum of 6 tests per day
* Result sent to registered emails

**TEST PAGE:**

 **Display a test with multiple-choice questions (MCQs)**

 **Allow students to select answers**

 **Include a countdown timer**

 **Submit answers to the backend**

 **Provide immediate feedback upon submission**

**Question Bank:**

All the questions to be uploaded unto iSchool OLA are based on selected class, subject and topics

Unlimited number of questions are set according to Class, Subject, topic.

The questions are uploaded under the various topics

* Format: Multiple choice

Theory

All the questions both multiple choice and theory questions in the App are based ONLY on selected class, subject and topics

Sample questions format:

Multiple choice and Theory questions

Example: Which of these is a state in Nigeria?

A.London

B.Benue

C.Ikeja

D.Atlantic

Answer: B

Theory:

Explain the health benefits of regular exercise

5. Test Configuration

o Time per test is 10 mins

o Number of questions per test: Maximum 10

o Randomization or fixed?: Randomization

o Can students retake tests?: No

6. Result Settings

o Email format/content for results to be emailed to students after taking test.

The result should indicate:

* Student name
* Date/time of test
* All the multiple choice questions attempted by the student
* The answer selected by student for each question
* The correct answer to each question
* All the theory questions attempted by the student
* The answer provided by the student

o Grading system

No grading is required for the multiple choice question because the both the correct answer and the answer originally chosen by the student will just be displayed on the result sheet.

For the theory, since the question is set based on an extract from a particular text, so the answers are already predetermined based on the content of the selected text on which the question was originally set.

The system can compare the answer provided by the student with the already predetermined answer.

Based on that, the system can assess and point out how much the answer provided aligns with the content of the text and predetermined answer in percentage

o Will results be stored and accessible later?: Possibly. A separate storage system can be inbuilt in the app for this purpose.

* Should students see correct answers after submission?: No

Authentication Method

o Email/password, phone number, school registration number?: There are 3 categories of end users:

Schools

Homes

Accredited Referrals

End users register result recipient emails

End user obtain generic School/Home/Accredited Referral IDs

End user generates individual usernames according to number of slots paid for

Only the last 3 digits of generic end user ID will change to indicate serial numbers of the students, according to the number of slots paid for

o Forgot password flow?: Not Applicable

Schools/Homes/Accredited Referrals cannot change passwords because these are automatically generated on the system to reflect State and school/home/accredited referral ID.

Students cannot change their password because it is a unique permanent serial identification automatically generated and attached to a School/Home/Accredited Referral.

This way, the system keeps track and record of entire user database

This way, if a subscription expires, user access is automatically gets blocked.

Dev stack:

 **Backend**: Django (Python 3 with pipenv) — serves both web and mobile apps

 **Frontend (Web)**: React + Bootstrap 5

 **Mobile App (Android/iOS)**: React Native (backend will expose APIs to be consumed)

**Prepare your documents for verification**

You will be required to upload two documents to help us and your business comply with legal and regulatory requirements.

**Government ID document**

We accept any of the following:

* Driver's License
* National Identity Card
* National Identity Number (NIN) Slip
* International passport

**Proof of address**

We accept any of the following:

* Statement of account from Bank or Other Financial Institution not more than 3 months old.
* Utility bill (electricity, water etc... not older than 3 months)
* A signed lease or tenancy agreement

**Business Documents for Limited Companies**

The following documents are required for Limited Companies:

* Certificate of Incorporation
* Memorandum & Articles of Association
* Shareholder Structure
* Board resolution confirming that approval has been given to open an account with us

user = CustomUser.objects.get(username="SC0655007221500")

With minor extensions (like email PDF delivery and enhanced theory scoring)

This is how the form should work on the TestPage.jsx.

1. Student select class level in a dropdown box with the title: “Select Your Class Level” form to select their class: JS1. JS2, JS3, SS1, SS2, SS3.
2. If the student selects JS1 for instance, let all subjects (15) related to JSS1 pop up. Among these subjects, if the student selects any of the subjects for instance Basic Science, let the topics under Basic science pop up
3. After selecting a topic, the 10minutes countdown begins and the preloaded questions will randomly pop up, one at a time
4. They can answer as many questions as they can within those 10 minutes.
5. After each question, they submit their answer . At the end of the 10 minutes, they finish test and result emailed to them.
6. Remember, Question model is already created.

# Dummy user

user = CustomUser.objects.get(email="gintason@c.com")

context = {

"student\_name": user.full\_name or user.email,

"date": datetime.now(),

"score": 85,

"total\_mcq": 50,

"correct\_mcq": 43,

"avg\_theory\_score": 8.5,

"answers": [], # You can fill with mock answers

"logo\_path": "/path/to/logo.png", # Optional: static logo path

}

# Generate PDF

html\_content = render\_to\_string("emails/test\_result\_report.html", context)

pdf\_file = BytesIO()

result = pisa.CreatePDF(src=html\_content, dest=pdf\_file)

pdf\_file.seek(0)

# Determine recipients

student\_email = user.email

parent\_email = getattr(user.registration\_group, 'email', None)

email\_subject = "iSchool Ola - Test Result"

email\_body = f"""Dear {context['student\_name']},

Please find attached your test result summary.

Best regards,

iSchool Ola Team"""

# Send to student and parent

for recipient in filter(None, [student\_email, parent\_email]):

try:

email = EmailMessage(

email\_subject,

email\_body,

'no-reply@ischool.ng', # or settings.DEFAULT\_FROM\_EMAIL

[recipient],

)

email.attach("Test\_Result.pdf", pdf\_file.getvalue(), "application/pdf")

email.send()

print(f"Email sent to: {recipient}")

except Exception as e:

logger.error(f"Failed to send result email to {recipient}: {e}")