Nigotis

API Documentation of Nigotis

version: v1
base_url = /api/v1

Types of users:

1. Super Admin (All Admin Users, Plans, Subscription)
2. Admin (All)
3. Employee
4. HR (Employee & Payroll Management, Run Payroll)
5. Sales (Client Management & Invoice Generation, Product Management)
6. Finance (Assets, Expenses, Income Management & Reports Generation)

Sr.	Name	URL	Method	Access	Input Dummy	Description
1.	List All Users	/user/all	GET	Super Admin	na	List all users with their respective data.

2.	Login	/user/login	POST	All	{
					"email": "superadmin@test.com",
					"password": "12345678"
					}
						Login api returns user data with token which is valid for 24 hour.

3.	detail data of user using id	/user/[id]	GET	Super admin	na	Return user data if user exist.
4.	signup	/user	POST	All	{ "name": "Test", "email": "test@test.com", "password": "12345678", "role": "admin"	Direct access for admin user who buys a plan and setup company. In this case company Id must be empty. Need Admin auth token for creating employee and manager role. In this case body must have companyId.

					}	For creating super admin it requires the auth token of super admin.

5.	User password update, also can update employeeId	/user	PUT	All	{
					"password": "12345678"
					}
						updating password.
						OR employeeId

6.	User Role Update	/user/role	PUT	Admin	{
					"userId": "6728f5177b9b7df2b008a043",
					"role": "hr"
					}
						Admin can assign the role to users of it’s company. [“admin”, "employee", "sales", "hr", "finance"]
						Company check added

7.	User Personal Info Update	/user/personal	PUT	All	{
					"title": "Mr.",
					"firstName": "John",
					"middleName": "Alexander",
					"lastName": "Doe",
					"address": "123 Elm Street Lahore Punjab Pakistan",
					"phone": "+92-321-1234567",
					"avatar": "https://example.com/avatars/johndoe.png",
					“joinDate”: “2020-12-31T00:00:00.000Z",
					}
						First upload the file to bucket and send the url of file in this API.

8.	List All Plans	/plan	GET	na	na	Na

9.	Create Plan	/plan	POST	Super Admin	{
					"type": "custom",
					"name": "Enterprise Powersoft19 Lahore",
					"screens": "100",
					"pricePerMonth": "150",
					"pricePerScreen": "2"
					}
						Basic, Pro, Premium, Custom

10.	Update Plan	/plan	PUT	Super Admin	{	Update plan....

					"_id": "672e2ab4edc99b920a64c549", "type": "custom", "name": "Enterprise Powersoft19 Lahore", "screens": "100", "price": "150", "pricePerScreen": "2" }	

11.	Delete Plan	/plan	DELETE	Super Admin	{
					"_id": "672e319f7f5404ea4d83c385"
					}
						na

12.	Buy Subscription	/plan/subscription	POST	Admin	{
					"planId": "672e2971a63d72e4a08f0264",
					"startDate": "2024-10-01T00:00:00.000Z",
					"endDate": "2024-12-31T00:00:00.000Z",
					"noOfMonths": 2,
					"transactionId": "TRX20241109123456",
					"totalScreens": 12,
					"totalFee": 60
					}
						This API must be called after payment is successful.
						Add Payment Check in API
						totalFee must be counted according to month + additionalScreens
						It returns subscription object which needs to be add in the object of user in frontend at the time of buying, after that when ever user logged in it will be returned along the token.

13.	List All subscription	/plan/subscription	GET	Super Admin	na	It also list the details of user who bought the subscription.

14.	Update/Renew subscription	/plan/subscription	PUT	Admin & Super Admin	{
					"planId": "672e2971a63d72e4a08f0264",
					"subscriptionId": "67373566665f0a1024b73419",
					"startDate": "2024-12-01",
					"endDate": "2025-03-01",
					"noOfMonths": 3,
					"transactionId": "TRX123456798",
					"totalScreens": 12,
					"totalFee": 90
					}
						As this api work well in renew and upgrade or downgrade of subscription.
						For just updating the no of screens, need separate api. /plan/subscription/screens PUT

15.	Delete Subscription	/plan/subscription	DELETE	Admin & super admin	{
					"_id": "672e2971a63d72e4a08f0264"
					}
						Not tested but seems to be fine.

16.	Buy screens	/plan/subscription/screens/buy	PUT	Admin & super admin	{
					"subscriptionId": "67373566665f0a1024b73419",
					"additionalScreens": "2",
					"additionalFee": "5",
					"notes": "optional"
					}
						Only add up no of screens

17.	Create Company	/company	POST	Admin	{
					"displayName": "Tech Innovators",
					"legalName": "Tech Innovators LLC",
					"type": "Technology",
					"ein": "12-3456789",
					"ssn": "123-45-6789",
					"logo": "https://example.com/logo.png",
					"address": "123 Innovation Drive, San Francisco, CA, USA",
					"phone": "+1-415-555-1234",
					"email": "contact@techinnovators.com"
					}
						Create a company

18.	Update company	/company	PUT	admin	{"displayName": "Tech Nest", "legalName": "Tech Nest LLC", "type": "Technology", "ein": "12-3456789", "ssn": "123-45-6789", "logo": "https://example.com/logo.png", "address": "123 Innovation Drive, San Francisco, CA, USA", "phone": "+1-415-555-1234", "email": "info@technest.com" }	Update a company
19.	List All companies	/company	GET	Super admin	na	List all companies data with admin details, subscription details.

20.	Delete company	/company	DELETE	Super admin & admin	{
					“companyId”: “iuwerwhdfj”
					}
						after finding copany we hace to delete it and it's all relative data, will implement this api later.

21.	List all users of specific company	/user	GET	admin	na	List all users of admin company

22.	Create payroll for an employee	/payroll	POST	admin	{
					"employeeId": "6728f5177b9b7df2b008a043",
					"salaryType": "fixed", // options: "fixed", "hourly"
					"salary": 10000,
					"bonus": 10,
					"tax": 6,
					"hourlyRate": 45,
					"overtimeHours": 0,
					"overtimeHourlyRate": 60
					}
						Payroll for an employee. Admin can create it’s own payroll as well.

23.	Update payroll	/payroll	PUT	admin	{ "payrollId": "67386d4c79cc0bfd99cb9759", "salaryType": "fixed", // options: "fixed", "hourly" "salary": 5000, "bonus": 5,	Admin can only update the payroll of an employee’s.

					"tax": 3, "hourlyRate": 0, "overtimeHours": 0, "overtimeHourlyRate": 30 }	
24.	List all payrolls of company	/payroll	GET	admin	na	na

25.	Run payroll	/payroll/runs	POST	Admin	{
					"employeeId": "67383ea8665f0a1024b734a0",
					"overtimeHours": 5,
					"status": "paid",
					"totalAmount": 2190,
					"noOfWorkingHours": 80,
					"attachments": [],
					"notes": "2000 total salary, 5% bonus, 3% tax, 150 for overtime ",
					"from": "2024-10-01T00:00:00.000Z",
					"to": "2024-11-01T00:00:00.000Z"
					}
						You have to calculate the total amount according to the payroll of an employee in frontend and pass value to backend.
						noOfWorkingHours in case of employee salary type is hourly based.
						It will create the expense according to run payrolls also set the status accordingly.
						in expense section it only shows if payroll is paid.

26.	List the run payrolls of specific employee	/payroll/runs?employeeId=4321	GET	Admin	na	na
27.	List all run payrolls of company	/payroll/runs/all	GET	Admin	na	na
28.	Update payroll run	/payroll/runs	PUT	Admin	{ "payrollRunsId": "67383e88665f0a1024b76543, "employeeId": "67383e88665f0a1024b73497", "overtimeHours": 5, "status": "paid", "totalAmount": 5250, "noOfWorkingHours": 80, "attachments": [], "notes": "5000 total salary, 5% bonus, 3% tax, 150 for overtime ",	Update the data of payroll runs and inform the user using mail and create a notification too.

					"from": "2024-10-01T00:00:00.000Z", "to": "2024-11-01T00:00:00.000Z" }	
29.	Delete run payroll	/payroll/runs?payrollRunsId=45234253	DELETE	Admin	na	na
30.	List notifications	/notification?type=all	GET	all	Type = all || new || read	
31.	Mark as viewed notification	/notification?_id=6739d55c90a1b25b845636a0	PUT	all	Query _id of notification	
32.	List All Assets	/company/asset	GET	admin	na	na

33.	Create asset	/company/asset	POST	admin	{
					"title": "Long Office Tables",
					"desc": "This is a description of the asset.",
					"quantity": 5,
					"date": "2024-10-01T00:00:00.000Z",
					"totalAmount": 2500.00
					}	

34.	Update the asset	/company/asset	PUT	admin	{
					"assetId": "673cc1b0073a77bee818772d",
					"title": "Office Chairs",
					"desc": "This is a description of the asset.",
					"quantity": 15,
					"date": "2024-10-01T00:00:00.000Z",
					"totalAmount": 1250.00
					}	

35.	Delete asset	/company/asset	DELETE	admin	{
					"assetId": "673cc231073a77bee8187747"
					}	

36.	List All Expenses	/company/expense	GET	admin	na	na

37.	Create Expense	/company/expense	POST	admin	{
					"title": "Sample Title 2",
					"desc": "Sample description of the asset.",
					"type": "others", // bill, payroll, others
					"totalAmount": 1500.50,
					"date": "2024-11-19T12:00:00Z",
					"from": "2024-11-01T08:30:00Z",
					"to": "2024-11-30T18:00:00Z"
					}	

38.	Update Expense	/company/expense	PUT	admin	{
					"expenseId": "673ccc3fd85c719ace5a9d66",
					"title": "Sample Title 3",
					"desc": "Sample description of the asset.",
					"type": "others", // bill, payroll, others
					"totalAmount": 1500.50,
					"date": "2024-11-19T12:00:00Z",
					"from": "2024-11-01T08:30:00Z",
					"to": "2024-11-30T18:00:00Z"
					}	

39.	Delete Expense	/company/expense	DELETE	admin	{
					"expenseId": "673cccbed85c719ace5a9d76"
					}	

40.	List All clients	/client	GET	Admin & sales	na	na
41.	Create client	/client	POST	Admin & sales	{ "firstName": "Faizan", "email": "client.two@example.com", "about": "Freelance digital marketer looking to improve reach.",	This will create client, personal info. And return the whole data of client created in this api.

					"title": "Mr.", "middleName": "", "lastName": "khan", "address": "77 Green Valley, Islamabad, Pakistan", "phone": "+92-344-0987654", "avatar": "https://example.com/avatars/bilal.jpg" , "joinDate": "2024-05-20" }	
42.	Update Client	/client	PUT	Admin & sales	clientId, email and about	For other updates like name, avatar, etc refer to personalInfo update API 7.
43.	Delete Client	/client	DELETE	Admin & sales	clientId	It will delete client, its produtcs, personal info. Invoice Delete if required.
44.	**NO NEED OF THESE API AFTER SALEPRODUCT**		POST		**NO NEED OF THESE API AFTER SALEPRODUCT**	**NO NEED OF THESE API AFTER SALEPRODUCT**

45.	**NO NEED OF THESE API AFTER SALEPRODUCT**	/client/product	PUT	Admin & manager	{
					"productId": "673d82192c161d8da368d7f3",
					"title": "Smartphone Stand 2.0",
					"desc": "Adjustable and foldable smartphone stand for desk use.",
					"quantity": 15,
					"price": 29.99
					}
						Na
						NO NEED OF THESE API AFTER SALEPRODUCT

46.	**NO NEED OF THESE API AFTER SALEPRODUCT**	/client/product	DELETE	Admin & manager	{
					"productId": "673d82192c161d8da368d7f3"
					}
						Na
						NO NEED OF THESE API AFTER SALEPRODUCT

47.	List All Invoices	/client/invoice	GET	Admin & sales	na	na

48.	Create invoice	/client/invoice	POST	Admin & sales	{
					"clientId": "673d7a0a2c161d8da368d7ac",
					"issueDate": "2024-11-20",
					"dueDate": "2024-12-20",
					"status": "pending",
					"tax": 10,
					"discount": 5,
					"paidAmount": 0,
					"items": [{ "productId": "677380e8c9db0bd395c0211f", "quantity": 2 }, { "productId": "6773801dc9db0bd395c0210a", "quantity": 1 }],
					}
						totalAmount is calculated in backend.
						income doc will also be created with mentioned status, here items array will be sent in body and api will create the saleproduct of each product.
						File part is remaining

49.	Update invoice	/client/invoice	PUT	Admin & sales	{
					"invoiceId": "673e0d963f23da2977e65ea2",
					"status": "paid",
					"paidAmount": 1055.95
					}
						File part is remaining remaining

50.	Delete Invoice	/client/invoice	DELETE	Admin & manager	{
					"invoiceId": "673e0d963f23da2977e65ea2"
					}	

51.	List All Product	/product	GET	Admin	na	List all products

52.	Create Product	/product	POST	Admin	{
					"companyId": "673788e3665f0a1024b73447",
					"name": "Website Maintenance",
					"desc": "web portal and frontnd backend Maintenance",
					"price": 600
					}	

53.	Update Product	/product	PUT	Admin	{
					"productId": "677380e8c9db0bd395c0211f",
					"name": "Marketing of store",
					"desc": "Marketing of store and store maintenance",
					"price": 400
					}	

54.	Delete Product	/product	DELETE	Admin	{
					"productId": "67738205c9db0bd395c02136"
					}	

55.	Add Income	/income	POST	Admin	{
					"type": "others",
					"totalAmount": 700,
					"date": "2024-12-25T12:00:00Z",
					"notes": "700 amount in. "
					}
						Type for frontend asset and others
						type invoice income will be created automatically if we have paid invoices.

56.	Update Income	/income	PUT	Admin	{
					"incomeId": "677a3a5d695ac68c178dff97",
					"type": "others",
					"totalAmount": 700,
					"date": "2024-12-25T12:00:00Z",
					"notes": "700 amount in. After update test. ",
					"status": "paid"
					}	

57.	Delete Income	/income	DELETE	Admin	{
					"incomeId": "677a3ac5695ac68c178dffab"
					}	

58.	List Income	/income?fromDate=2024-10-01&toDate=2024-12-31	GET	admin	na	1. Without dates list all incomes, 2. fromDate till today 3. toDate list till mentioned date 4. both for specific period
59.	List All Sold Products	/sale-product	GET	Sales	na	na