System Design Document

Friday October 22, 2021

Sprint 2

Krutik Tejalkumar Shah
Mark (LiangShu) Chen
Steven Hans Limantoro
Tanzim Ahmed
Tony Chen
Pasa Aslan
Payam Yektamaram

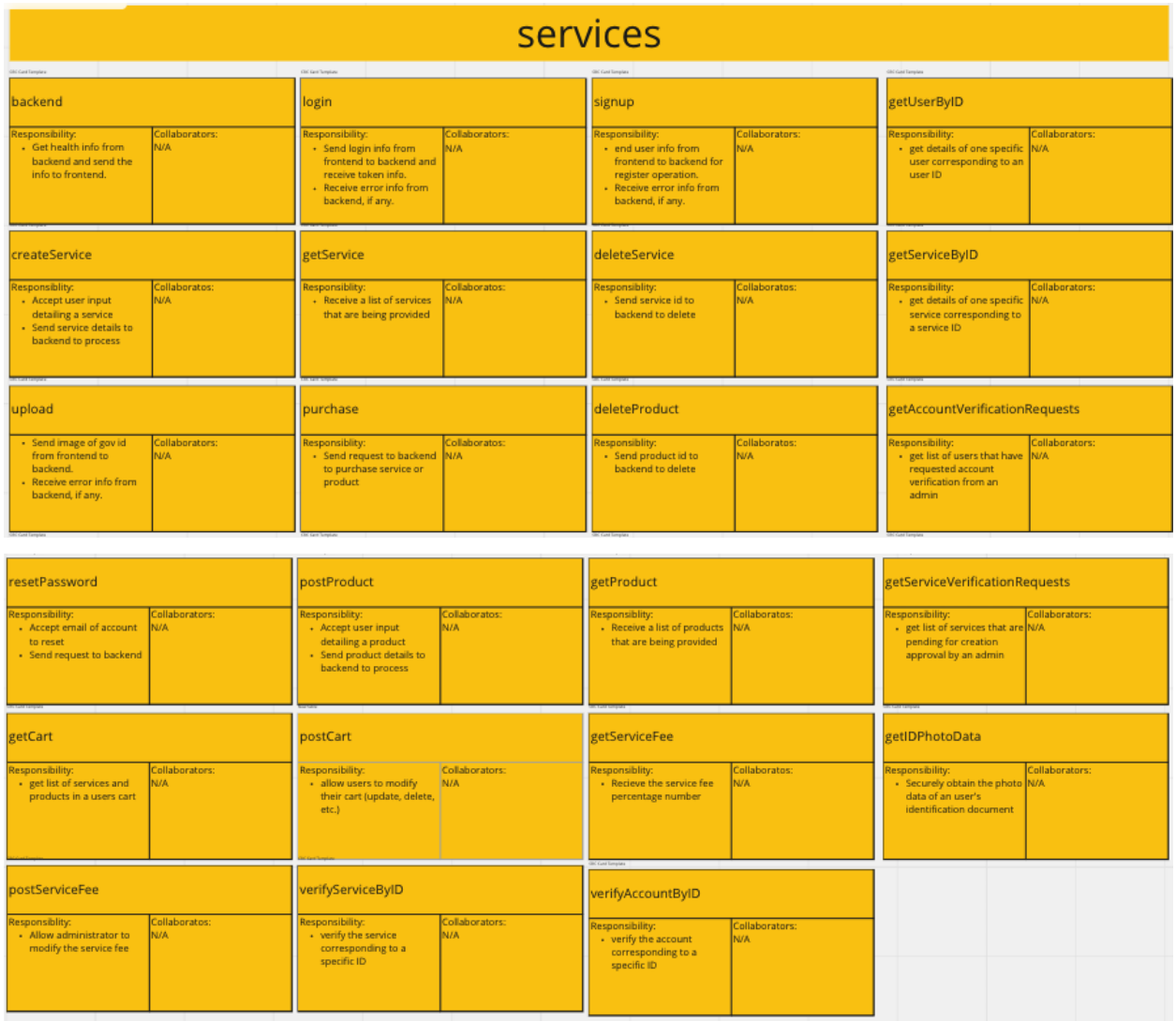
Table of Contents

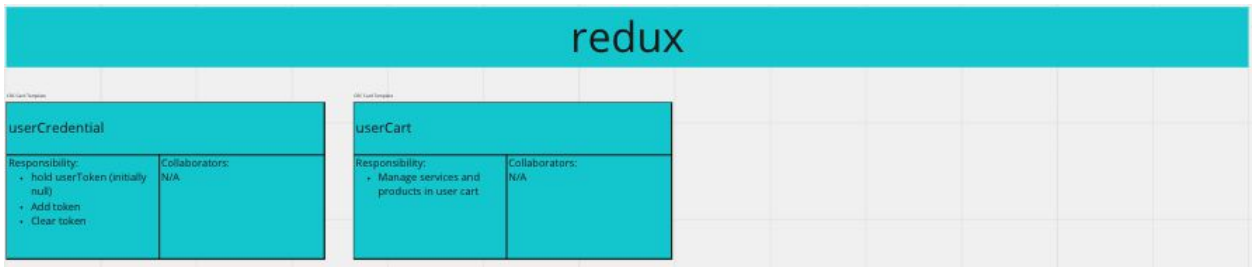
CRC Cards **2**
Frontend 2
Backend 4

System Architecture Diagram (Node.js Layered Arch) **6**
Frontend 6
Architecture 6
Backend 7
Server Endpoints & Descriptions 7
Database Schema Models 8
Architecture 9

CRC Cards

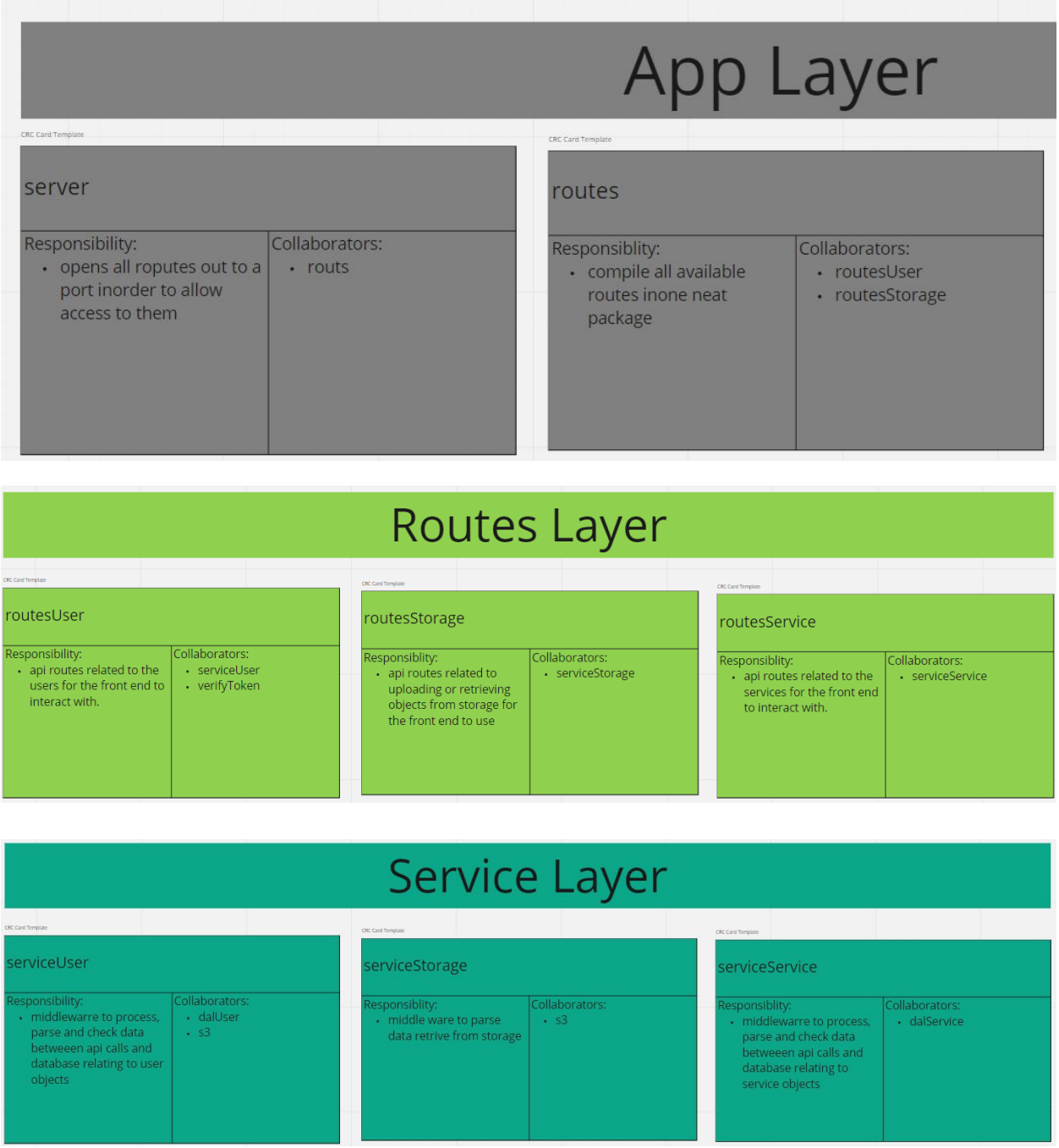
Frontend





screens
SignInScreen
Responsibility:
Allow user to input email and password
Validate user credentials
Authenticate user session
Link to Sign-Up.
Link to Forgot-Password.
Collaborators:
login
userCredential
SignUpScreen
Responsibility:
Accept email, username, and password to register new user
Validate user input
Link to SignInScreen
Collaborators:
signup
ForgotPasswordScreen
Responsibility:
Accept email of account to reset password
Collaborators:
forgotPassword
HomeTabScreen
Responsiblity:
Provide detailed search functionality for services
Collaborators:
N/A
Services
Responsibility:
List available services
Provide filtering functionality
Allow verified users to post their own service
Allow user to view a specific service in more detail
Allow user to delete their own service
Allow user to purchase service
Collaborator:
Services
CreateServiceModalContact
StoreTabScreen
Responsibility:
List available product
Provide filtering functionality
Allow user to view a specific product in more detail
Allow admin to delete product
Allow user to purchase product
Collaborators:
postProduct
getProduct
deleteProduct
purchase
CartTabScreen
Responsibility:
List services and products in the users cart
Allow user to manage their cart
Collaborators:
getCart
postCart
userCart
ServiceFeeModal
Responsibility:
Display current service fee
Allow administrator to update service fee
Collaborators:
getServiceFee
postServiceFee
ServiceApprovalModal
Responsibility:
Display the details of the service pending approval
Allow administrator to approve/deny the service creation
Collaborators:
useAppSelector
verifyServiceByID
CreateServiceModalDescription
Responsibility:
Allow users to enter service name, description and price
Give two options: continue to enter contact information, and cancel to return services screen.
Collaborator:
Services
CreateServiceModalContact
ResetPasswordScreen
Responsiblity:
Accept verification code and new password
Collaborators:
resetPassword
Frontend CRC Cards, Online Whiteboard for Visual Collaboration
Responsibility:
Allow users to enter contact number, country, city, postal code, and address.
Give two options: request service verification, and cancel to return services screen.
Collaborator:
Services
CreateServiceModalDescription
createService
AccountTabScreen
Responsibility:
List user info
Allow user to upload government id
Allow user to log out
Collaborators:
upload
NotificationsScreen
Responsibility:
Display notifications for a specific user
Collaborators:
getUserInfoByID
getAccountVerificationRequests
getServiceInfoByID
getServiceVerificationRequests
VerificationApprovalModal
Responsibility:
Display the details of the user pending verification approval
Allow administrator to approve/deny the verification of the user
Collaborators:
useAppSelector
verifyUserByID
getIDPhotoData

Backend


App Layer
CRC Card Template
server
Responsibility:
• opens all roputes out to a port inorder to allow access to them
Collaborators:
• routs
CRC Card Template
routes
Responsiblity:
• compile all available routes inone neat package
Collaborators:
• routesUser
• routesStorage
Routes Layer
CRC Card Template
routesUser
Responsibility:
• api routes related to the users for the front end to interact with.
Collaborators:
• serviceUser
• verifyToken
CRC Card Template
routesStorage
Responsiblity:
• api routes related to uploading or retrieving objects from storage for the front end to use
Collaborators:
• serviceStorage
CRC Card Template
routesService
Responsiblity:
• api routes related to the services for the front end to interact with.
Collaborators:
• serviceService
Service Layer
CRC Card Template
serviceUser
Responsibility:
• middlewarre to process, parse and check data betweeen api calls and database relating to user objects
Collaborators:
• dalUser
• s3
CRC Card Template
serviceStorage
Responsiblity:
• middle ware to parse data retrive from storage
Collaborators:
• s3
CRC Card Template
serviceService
Responsiblity:
• middlewarre to process, parse and check data betweeen api calls and database relating to service objects
Collaborators:
• dalService

Data Access Layer

dalUser	
Responsibility: • upload and query user related objects from the database • util function related to user objects	Collaborators: modelUser modelUserCredentials modelUserVerificationRequests

dalService	
Responsibility: • upload and query service related objects from the database • util function related to service objects	Collaborators: modelService modelVerificationRequests

Models

modelUser	
Responsibility: • define User object schema	Collaborators: N/A

modelUserCredentials	
Responsibility: • define UserCredential object schema	Collaborators: N/A

modelUserVerificationRequest	
Responsibility: • Define UserVerificationRequest object schema	Collaborators: N/A

modelService	
Responsibility: • Define Service object schema	Collaborators: N/A

modelServiceVerificationRequest	
Responsibility: • Define ServiceVerificationRequest object schema	Collaborators: N/A

Utils

s3	
Responsibility: • upload and retrive objectds from s3 bucket	Collaborators: N/A

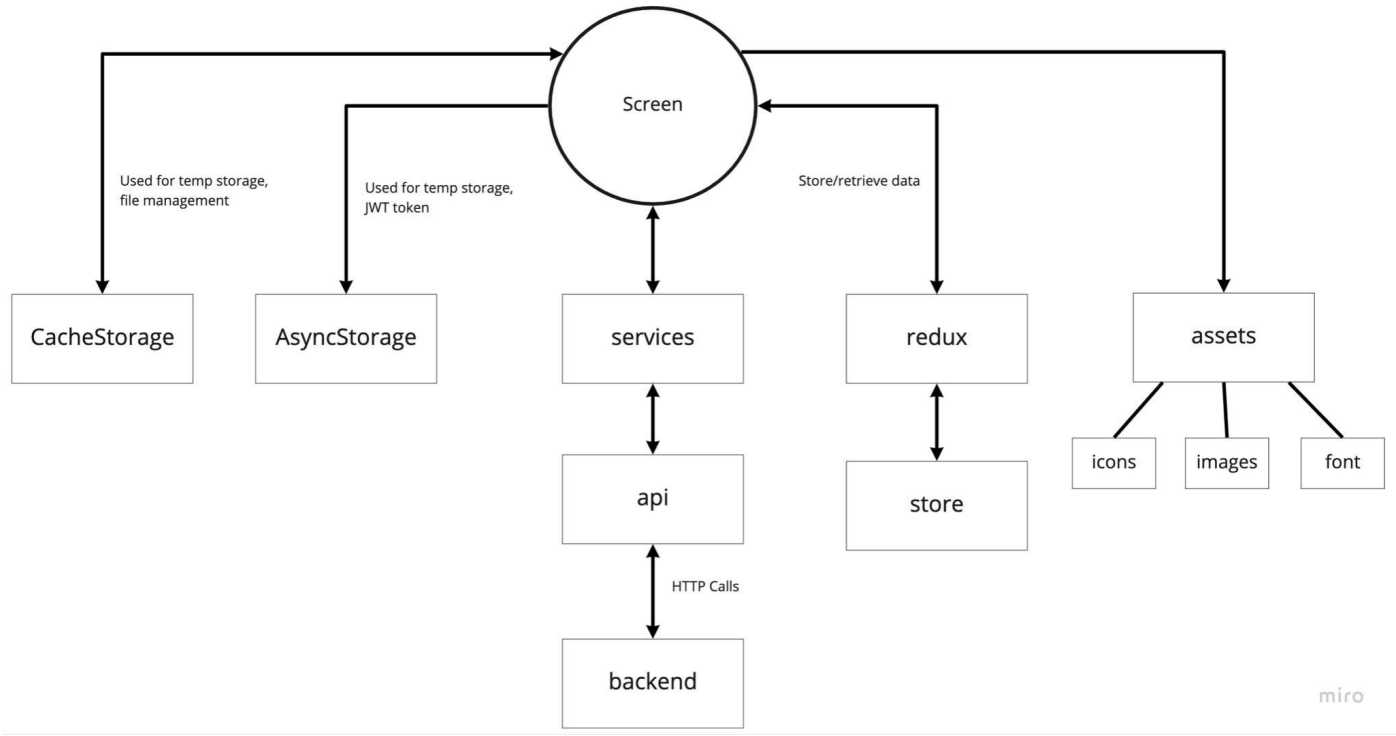
verifyToken	
Responsibility: • verify session token	Collaborators: N/A

emailConfig	
Responsibility: • create email templates	Collaborators: N/A

System Architecture Diagram (*Node.js Layered Arch*)

Frontend

Architecture



Backend

Server Endpoints & Descriptions

End Points

Summary	Path	Auth Required	Verification level Required	Method	Payload	Response
Sprint 1						

Register	/api/v1/users/register	No	None	POST	{
					email: email,
					username: user,
					password: password
					}
						USER

Login	/api/v1/users/login	No	None	POST	{
					email: email,
					password: password
					}
						USER

Forgot Password	/api/v1/forgot-password/{email}	No	None	POST	email: email	None

Reset Password	/api/v1/reset-password/{email}	No	None	POST	{
					encryptedEmail: email,
					encryptedOTPId: id,
					otp: code,
					password: password
					}
						USER

Update Account Details	/api/v1/users/self	Yes	User	PUT	{
					first_name: f_name,
					last_name: l_name,
					address: address,
					phone: phone
					}
						USER

Request User Verfication	/api/v1/users/self/request-verification	Yes	Unverified User	POST	gov-id: png/jpg	None
Retrieve User Verification	/api/v1/users/verification-request	Yes	Owner/Admin	GET	Query Params: limit: int skip: int	List[Verification_request]

Approve User Verification Request	/api/v1/users/verification-request	Yes	Owner/Admin	PUT	{
					user_id: user_id,
					approved: boolean
					}
						None

Get Account Details	/api/v1/users/self	Yes	User	GET	None	USER
Get Account Details By Id	/api/v1/users/{user_id}	No	None	GET	None	USER
Get Image	/api/v1/storage/media/{image_key}	No	None	GET	None	png/jpg

Sprint 2						

Post service (request to verify service)	/api/v1/services/request-verification	Yes	Verified User	POST	{
					service_name: name,
					service_description: description,
					service_price: price,
					contact_number: number,
					country: country,
					city: city,
					postal_code: code,
					address: address
					}
						None

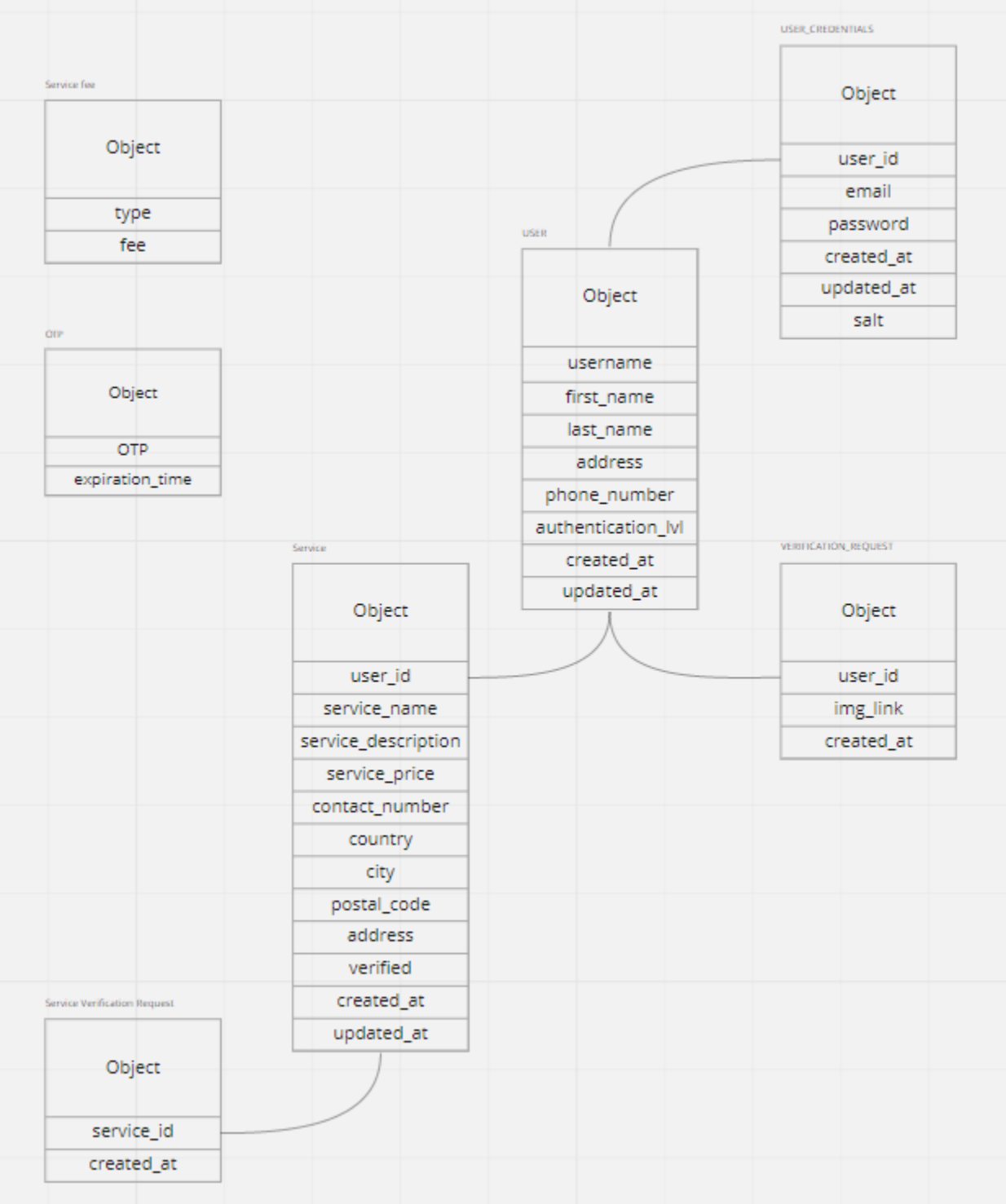
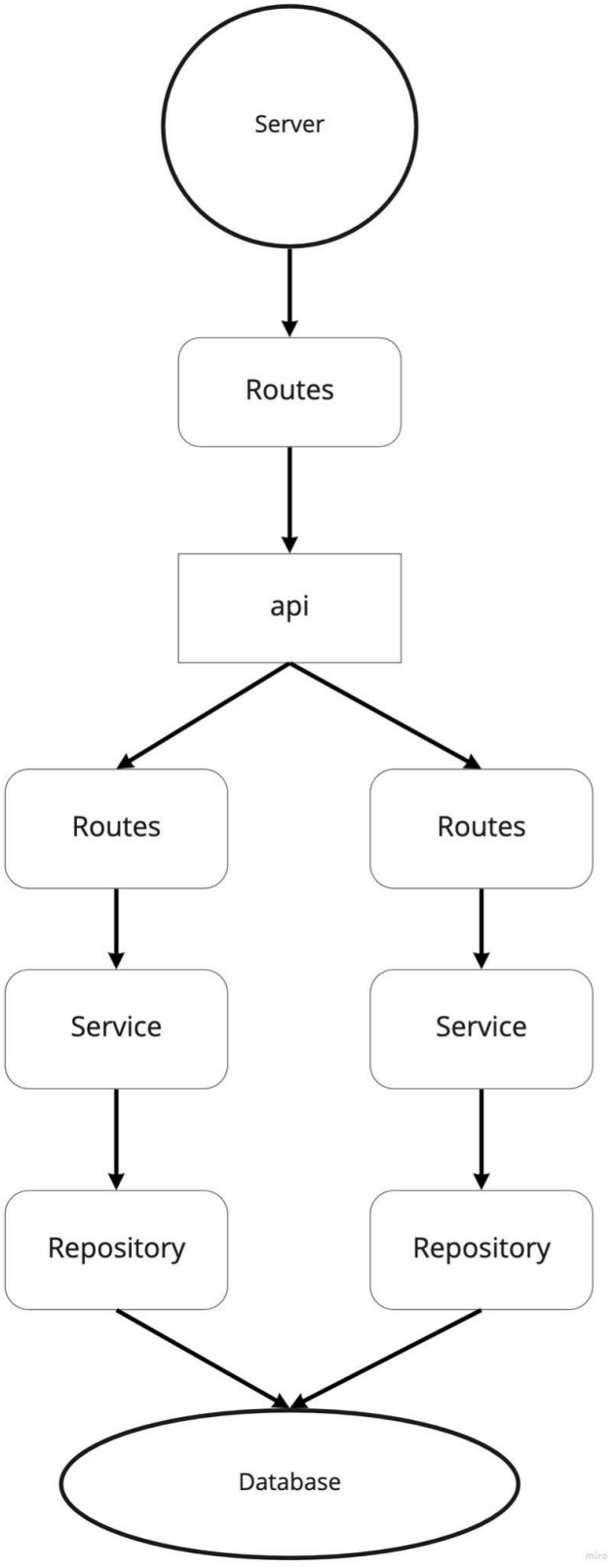
Retrieve Service Verification	/api/v1/services/verification-request	Yes	Owner/Admin	GET	Query Params: limit: int skip: int	List[Service_verification_request]

Approve Service Verification Request	/api/v1/services/verification-request	Yes	Owner/Admin	PUT	{
					service_id: service_id,
					approved: boolean
					}
						None

Get Service Details	/api/v1/services/{serviceId}	No	None	GET	None	Service
Get Verified Services	/api/v1/services/verified	No	None	GET	Query Params: limit: int skip: int	List[Service]
Get Service Fees	/api/v1/services/getFees	Yes	Owner/Admin	GET	None	Fee

Update Service Fees	/api/v1/services/updateFees	yes	Owner/Admin	PUT	{
					fee: Number
					}
						None

Database Schema Models



Architecture