System Design Document

Friday November 5, 2021

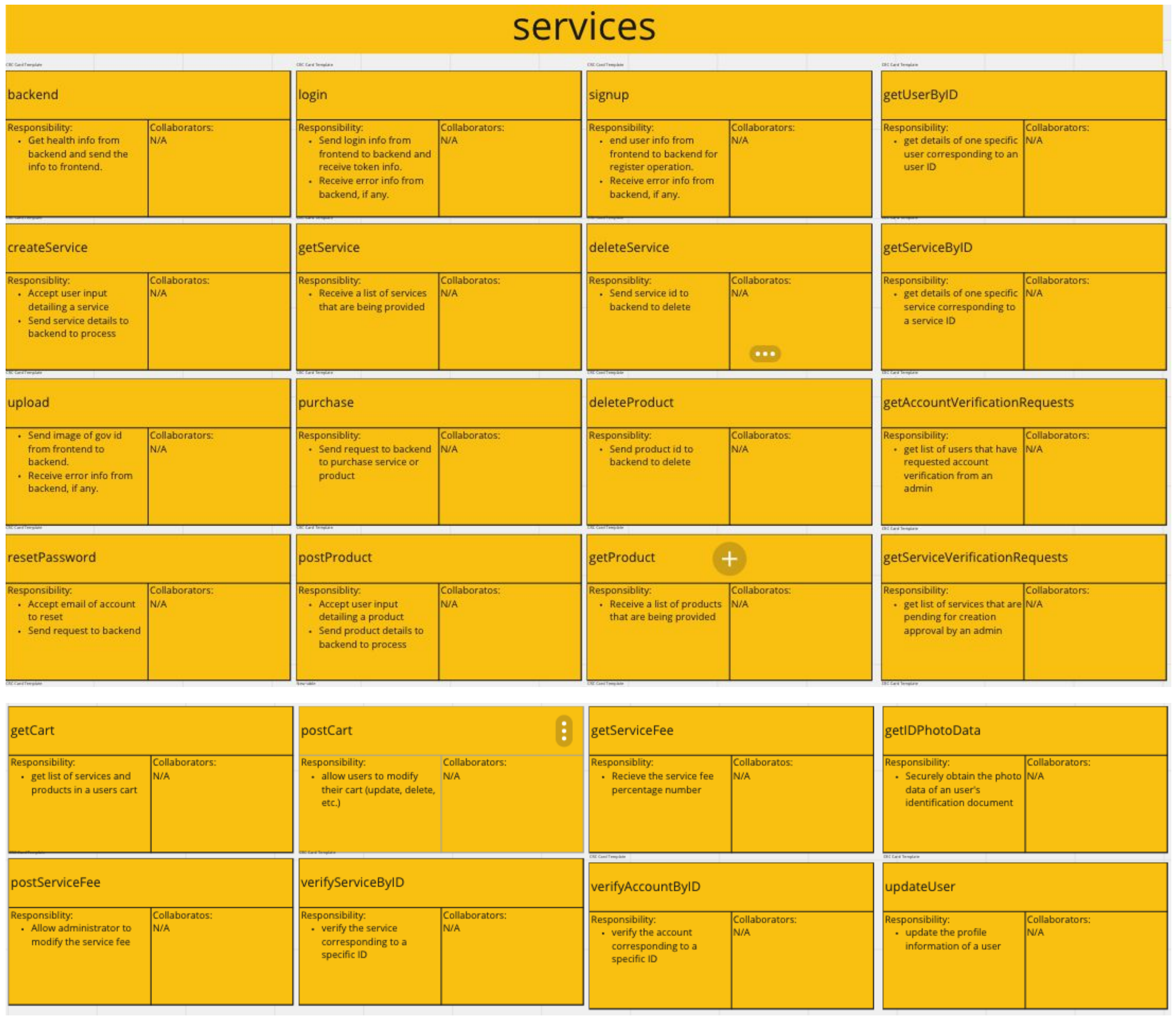
Sprint 3

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

Table of Contents

CRC Cards

Frontend



services

backend	
Responsibility: • Get health info from backend and send the info to frontend.	Collaborators: N/A

login	
Responsibility: • Send login info from frontend to backend and receive token info. • Receive error info from backend, if any.	Collaborators: N/A

signup	
Responsibility: • end user info from frontend to backend for register operation. • Receive error info from backend, if any.	Collaborators: N/A

getUserByID	
Responsibility: • get details of one specific user corresponding to an user ID	Collaborators: N/A

createService	
Responsiblity: • Accept user input detailing a service • Send service details to backend to process	Collaboratos: N/A

getService	
Responsiblity: • Receive a list of services that are being provided	Collaboratos: N/A

deleteService	
Responsiblity: • Send service id to backend to delete	Collaboratos: N/A

getServiceByID	
Responsibility: • get details of one specific service corresponding to a service ID	Collaborators: N/A

upload	
• Send image of gov id from frontend to backend. • Receive error info from backend, if any.	Collaborators: N/A

purchase	
Responsibility: • Send request to backend to purchase service or product	Collaboratos: N/A

deleteProduct	
Responsiblity: • Send product id to backend to delete	Collaboratos: N/A

getAccountVerificationRequests	
Responsibility: • get list of users that have requested account verification from an admin	Collaborators: N/A

resetPassword	
Responsibility: • Accept email of account to reset • Send request to backend	Collaborators: N/A

postProduct	
Responsibility: • Accept user input detailing a product • Send product details to backend to process	Collaboratos: N/A

getProduct	
Responsiblity: • Receive a list of products that are being provided	Collaboratos: N/A

getServiceVerificationRequests	
Responsibility: • get list of services that are pending for creation approval by an admin	Collaborators: N/A

getCart	
Responsibility: • get list of services and products in a users cart	Collaborators: N/A

postCart	
Responsibility: • allow users to modify their cart (update, delete, etc.)	Collaborators: N/A

getServiceFee	
Responsiblity: • Recieve the service fee percentage number	Collaboratos: N/A

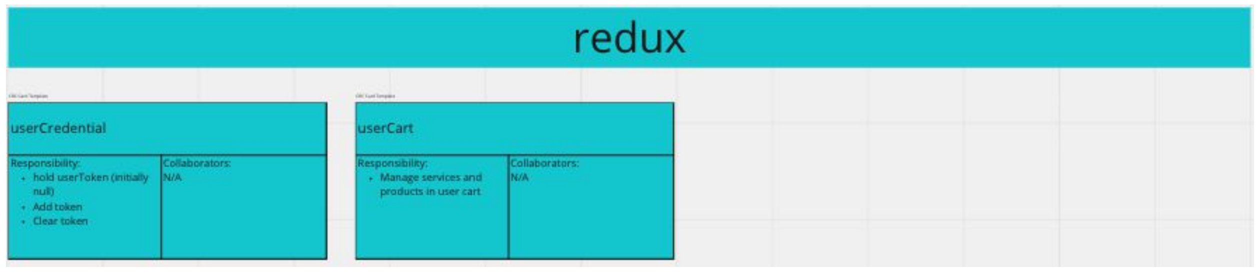
getIDPhotoData	
Responsibility: • Securely obtain the photo data of an user's identification document	Collaborators: N/A

postServiceFee	
Responsiblity: • Allow administrator to modify the service fee	Collaboratos: N/A

verifyServiceByID	
Responsibility: • verify the service corresponding to a specific ID	Collaborators: N/A

verifyAccountByID	
Responsibility: • verify the account corresponding to a specific ID	Collaborators: N/A

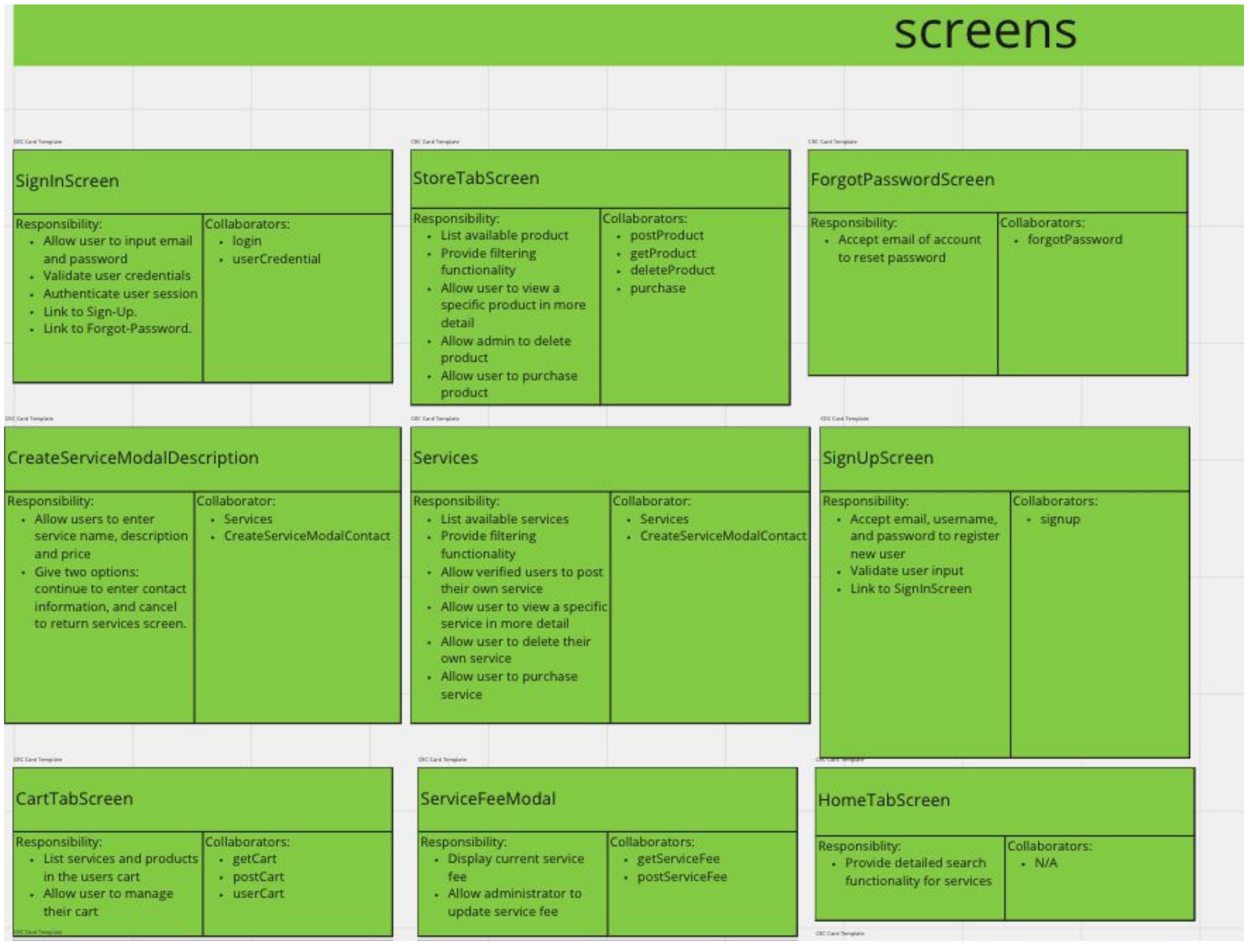
updateUser	
Responsibility: • update the profile information of a user	Collaborators: N/A



redux

userCredential	
Responsibility: • hold userToken (initially null) • Add token • Clear token	Collaborators: N/A

userCart	
Responsibility: • Manage services and products in user cart	Collaborators: N/A



screens

SignInScreen

Responsibility:	Collaborators:
• Allow user to input email and password • Validate user credentials • Authenticate user session • Link to Sign-Up. • Link to Forgot-Password.	• login • userCredential

StoreTabScreen

Responsibility:	Collaborators:
• 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	• postProduct • getProduct • deleteProduct • purchase

ForgotPasswordScreen

Responsibility:	Collaborators:
• Accept email of account to reset password	• forgotPassword

CreateServiceModalDescription

Responsibility:	Collaborator:
• Allow users to enter service name, description and price • Give two options: continue to enter contact information, and cancel to return services screen.	• Services • CreateServiceModalContact

Services

Responsibility:	Collaborator:
• 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	• Services • CreateServiceModalContact

SignUpScreen

Responsibility:	Collaborators:
• Accept email, username, and password to register new user • Validate user input • Link to SignInScreen	• signup

CartTabScreen

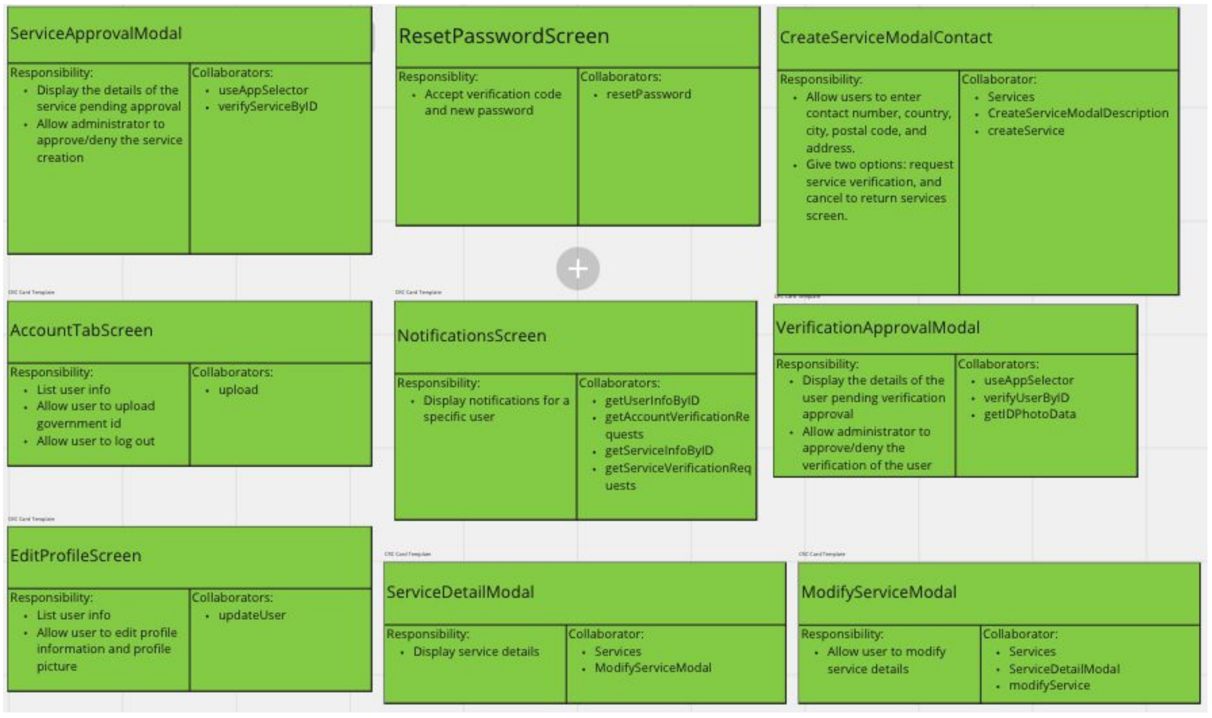
Responsibility:	Collaborators:
• List services and products in the users cart • Allow user to manage their cart	• getCart • postCart • userCart

ServiceFeeModal

Responsibility:	Collaborators:
• Display current service fee • Allow administrator to update service fee	• getServiceFee • postServiceFee

HomeTabScreen

Responsiblity:	Collaborators:
• Provide detailed search functionality for services	• N/A



ServiceApprovalModal

Responsibility:	Collaborators:
• Display the details of the service pending approval • Allow administrator to approve/deny the service creation	• useAppSelector • verifyServiceByID

ResetPasswordScreen

Responsibility:	Collaborators:
• Accept verification code and new password	• resetPassword

CreateServiceModalContact

Responsibility:	Collaborator:
• Allow users to enter contact number, country, city, postal code, and address. • Give two options: request service verification, and cancel to return services screen.	• Services • CreateServiceModalDescription • createService

AccountTabScreen

Responsibility:	Collaborators:
• List user info • Allow user to upload government id • Allow user to log out	• upload

NotificationsScreen

Responsibility:	Collaborators:
• Display notifications for a specific user	• getUserInfoByID • getAccountVerificationRequests • getServiceInfoByID • getServiceVerificationRequests

VerificationApprovalModal

Responsibility:	Collaborators:
• Display the details of the user pending verification approval • Allow administrator to approve/deny the verification of the user	• useAppSelector • verifyUserByID • getIDPhotoData

EditProfileScreen

Responsibility:	Collaborators:
• List user info • Allow user to edit profile information and profile picture	• updateUser

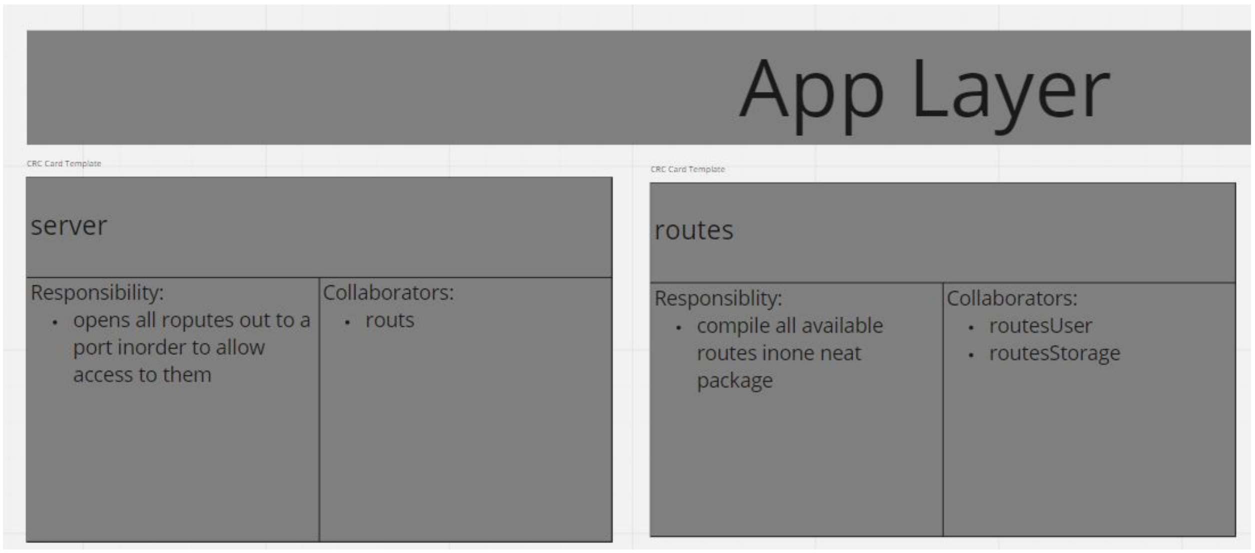
ServiceDetailModal

Responsibility:	Collaborator:
• Display service details	• Services • ModifyServiceModal

ModifyServiceModal

Responsibility:	Collaborator:
• Allow user to modify service details	• Services • ServiceDetailModal • modifyService

Backend



App Layer

CRC Card Template

server

Responsibility:
- opens all roputes out to a port inorder to allow access to them

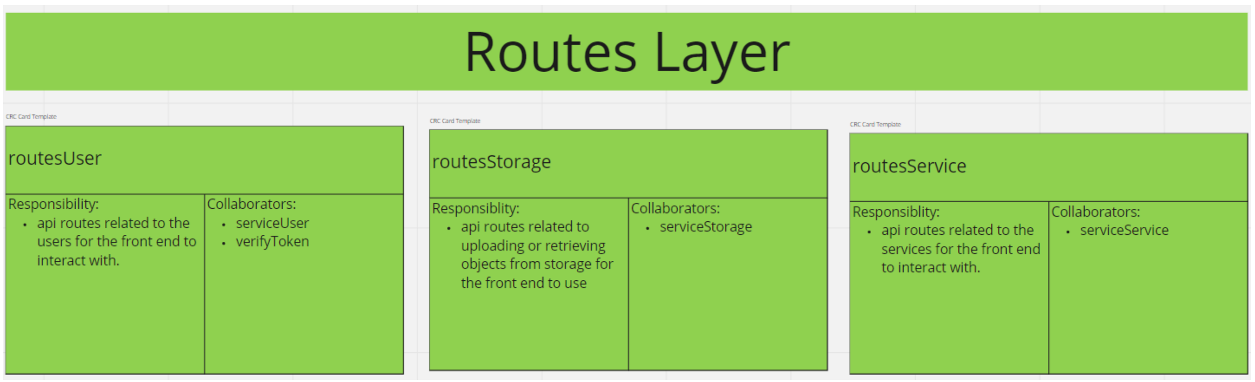
Collaborators:
- routs

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

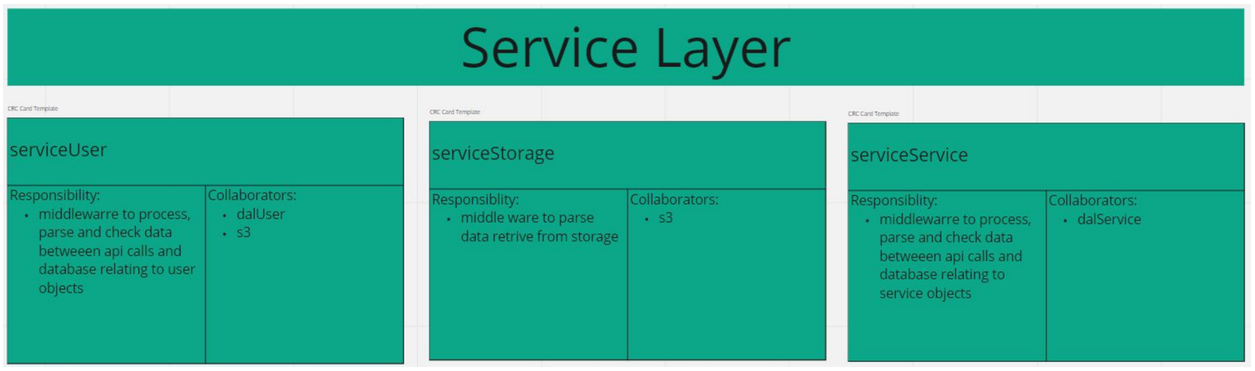
Collaborators:
- serviceStorage

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

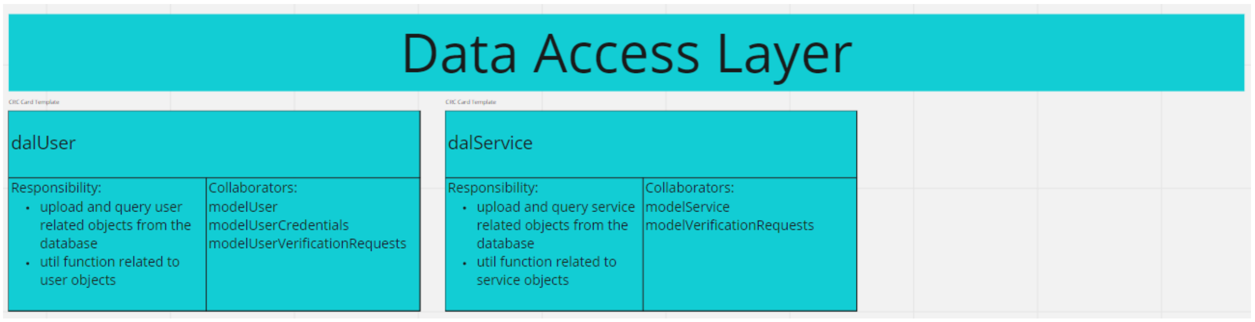
Collaborators:
- s3

CRC Card Template

serviceService

Responsiblity:
- middlewarre to process, parse and check data betweeen api calls and database relating to service objects

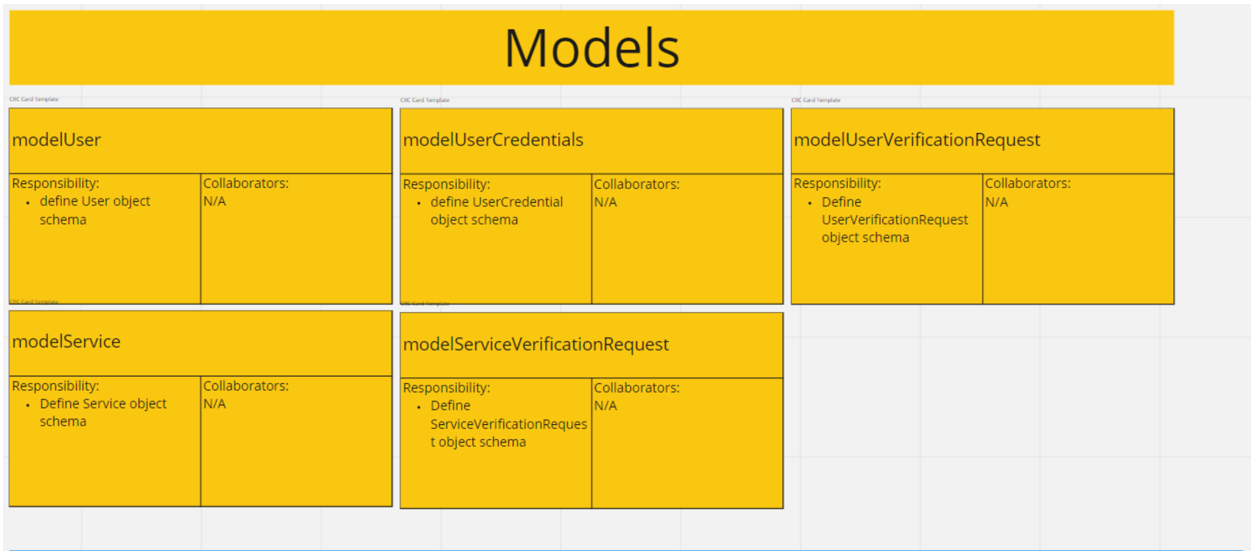
Collaborators:
- dalService



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

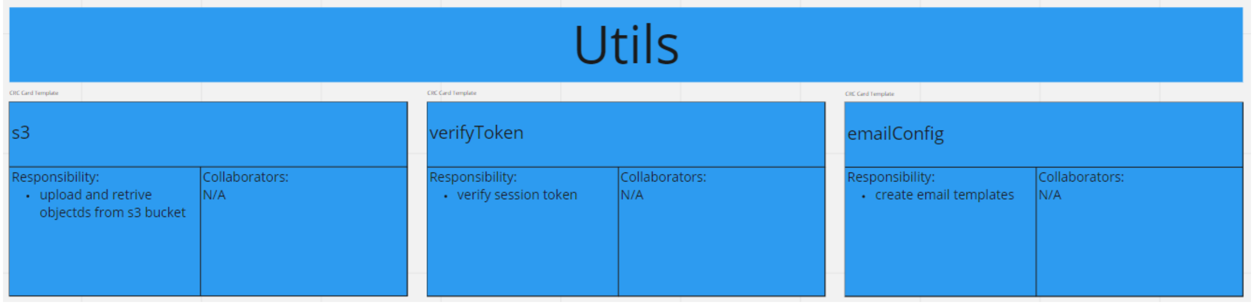
modelUser	
Responsibility: • define User object schema	Collaborators: N/A

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

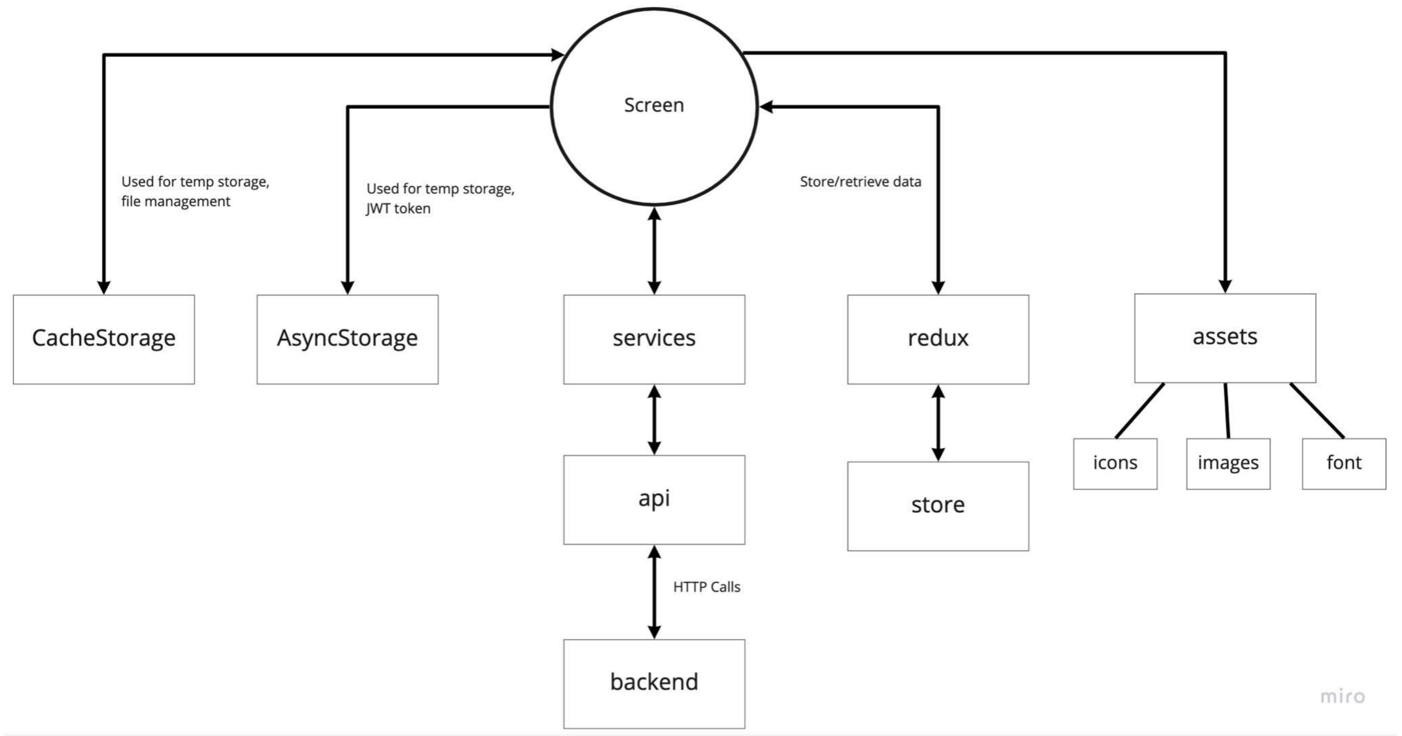
verifyToken	
Responsibility: • verify session token	Collaborators: N/A

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

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

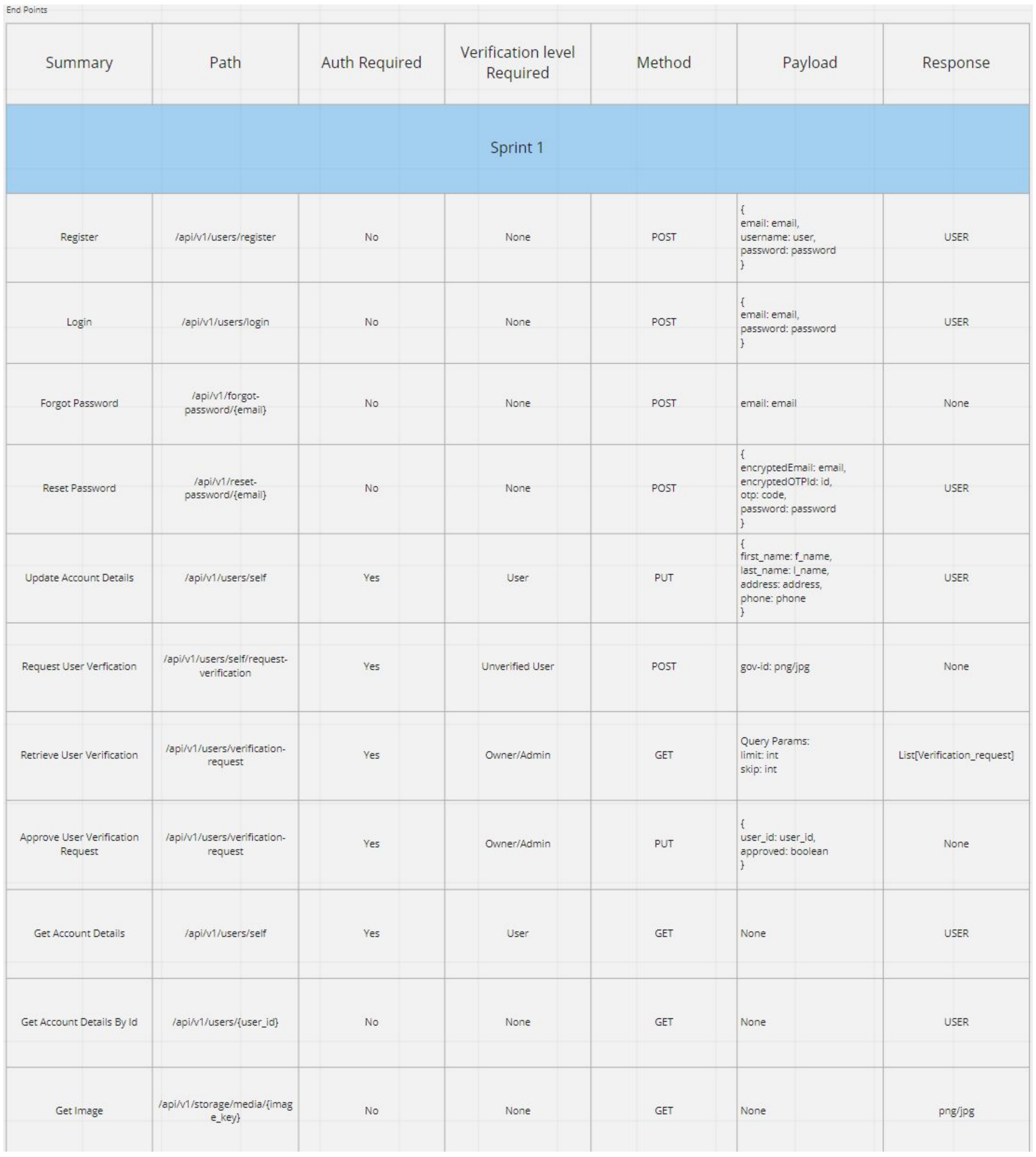
Frontend

Architecture



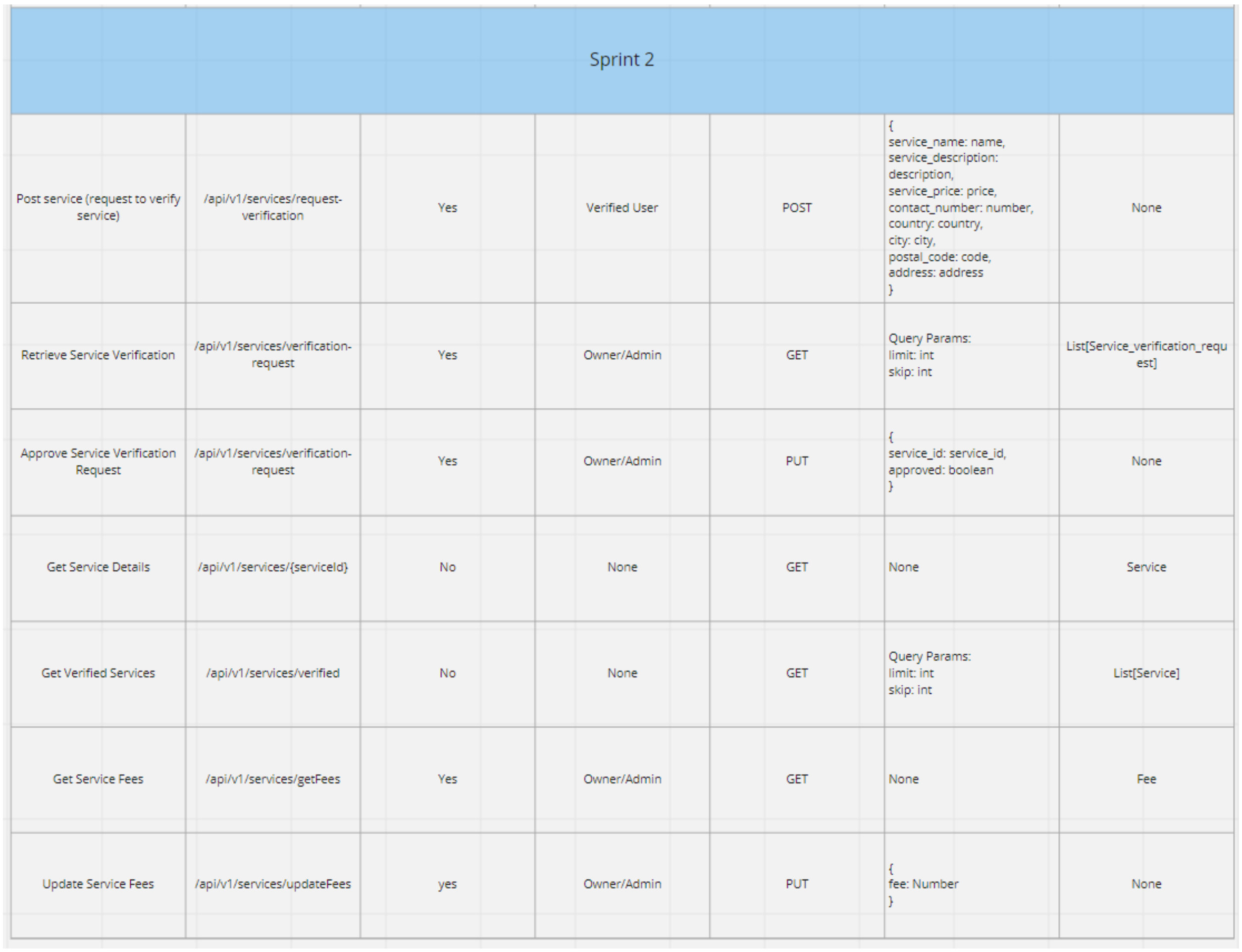
Backend

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						

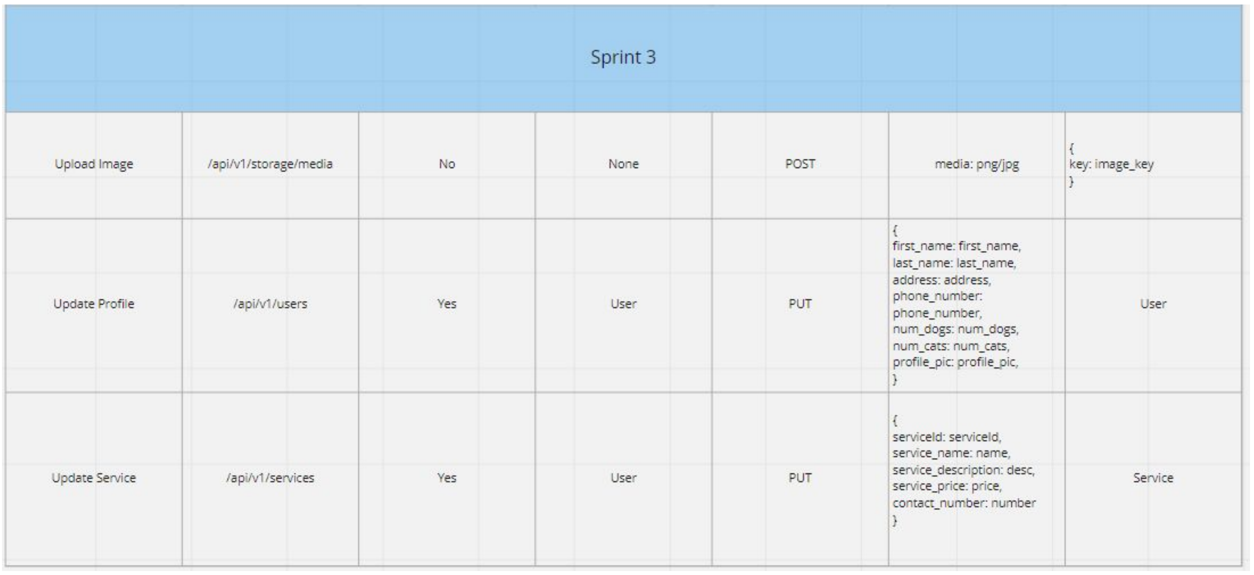
Post service (request to verify service)	/api/v1/services/request-verification	Yes	Verified User	POST	{
					service_name: name,
					service_description: description,
					service_price: price,
					contact_number: number,
					country: country,
					city: city,
					postal_code: code,
					address: 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



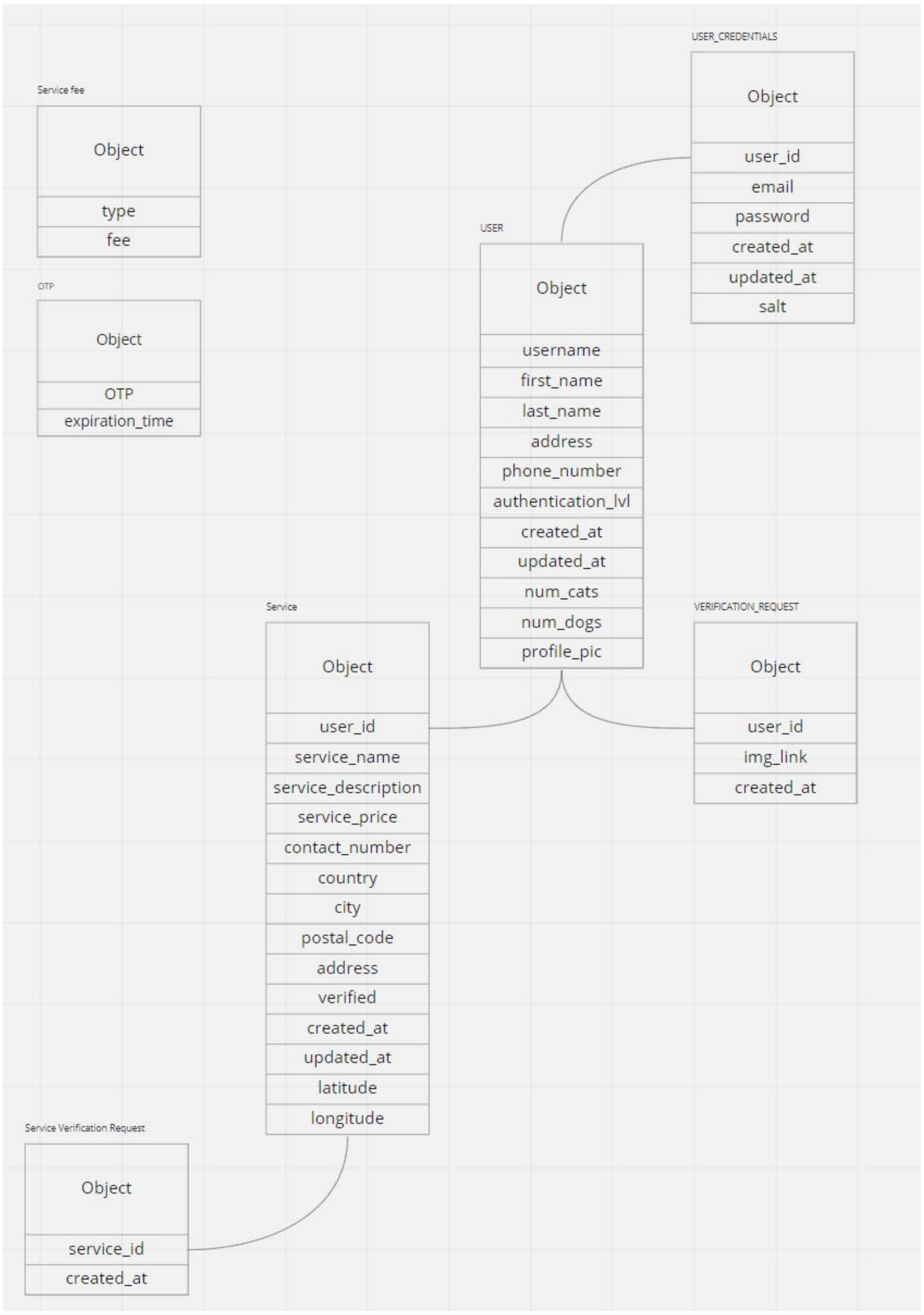
Sprint 3						

Upload Image	/api/v1/storage/media	No	None	POST	media: png/jpg	{
						key: image_key
						}

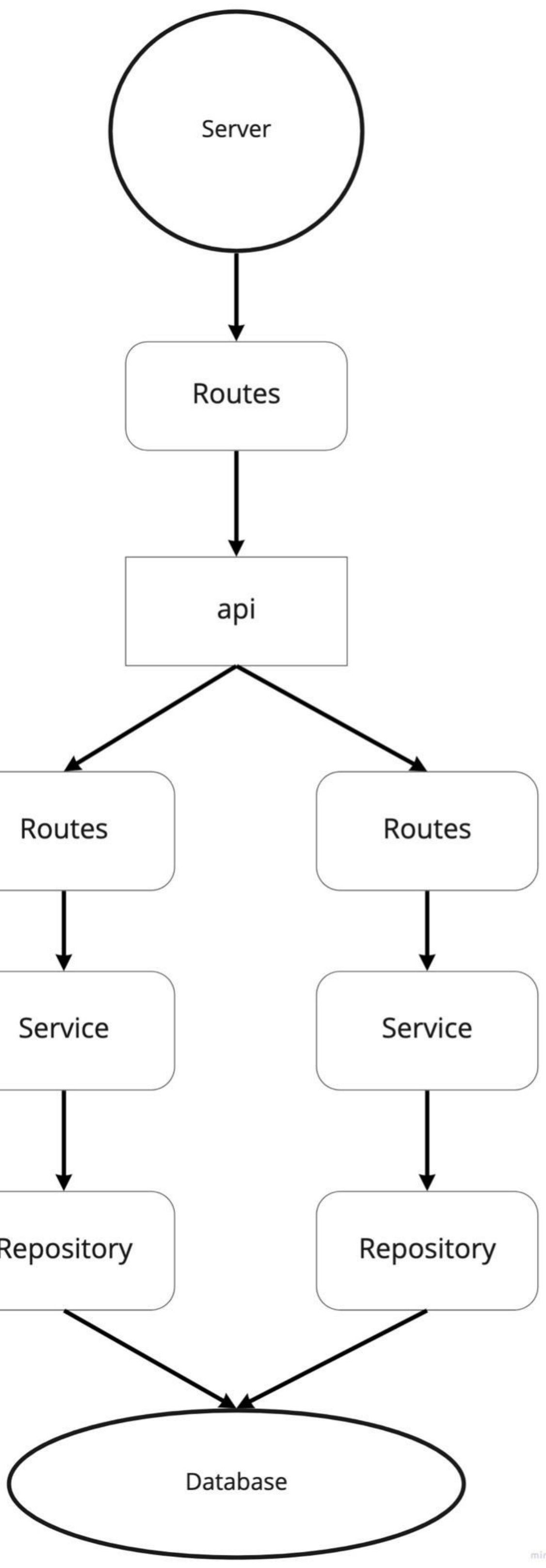
Update Profile	/api/v1/users	Yes	User	PUT	{
					first_name: first_name,
					last_name: last_name,
					address: address,
					phone_number: phone_number,
					num_dogs: num_dogs,
					num_cats: num_cats,
					profile_pic: profile_pic,
					}
						User

Update Service	/api/v1/services	Yes	User	PUT	{
					serviceId: serviceId,
					service_name: name,
					service_description: desc,
					service_price: price,
					contact_number: number
					}
						Service

Database Schema Models



Architecture


Server
Routes
api
Routes
Routes
Service
Service
Repository
Repository
Database
miro