



# Software Architecture & Development: **PetPal**

Master of Science  
Applied Computer Science

Kamal Kumar Sardiwal



# Introduction

01



# PetPal

Our app provides everything a pet needs in one convenient place. From adoption and events to vet bookings and daycare, we've got it all.

## Features:



Adoption services



Pet events



Pet Services - (Veterinary and Grooming Centers)



Lost and found

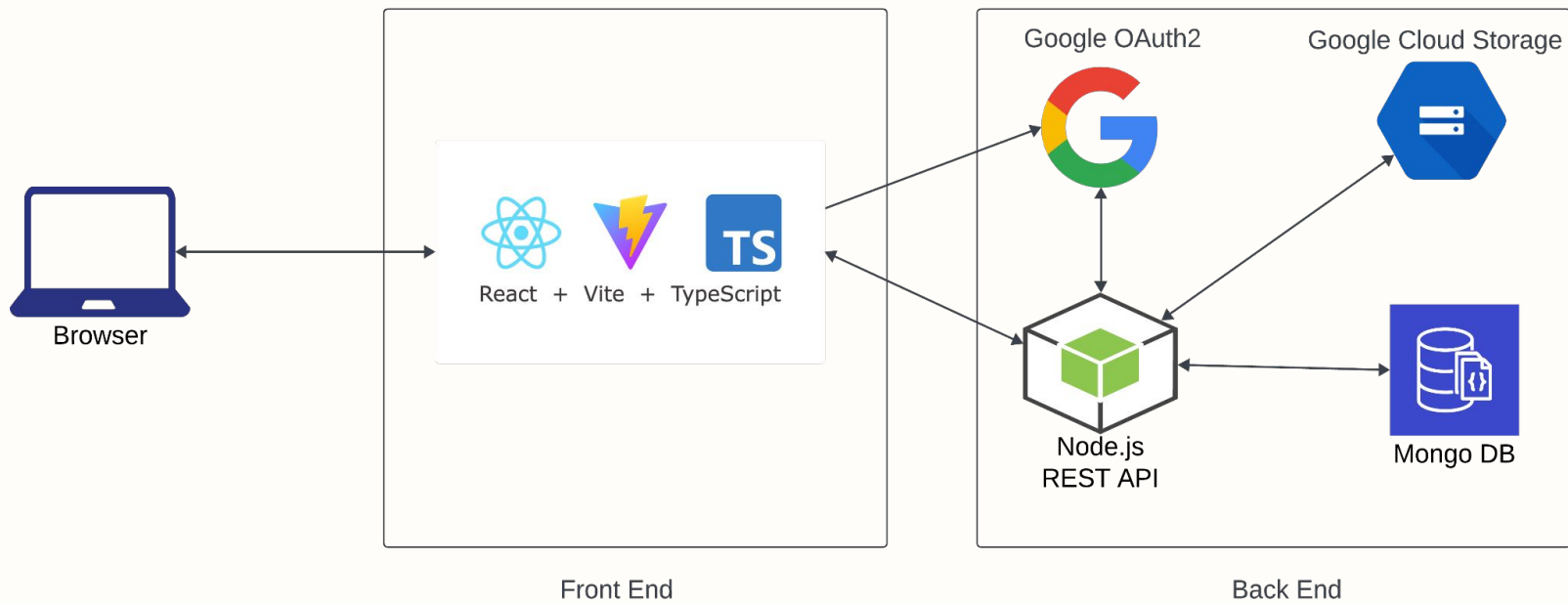


Pet Facilities - (Daycares and Caretakers)

# Software Architecture Diagram

02



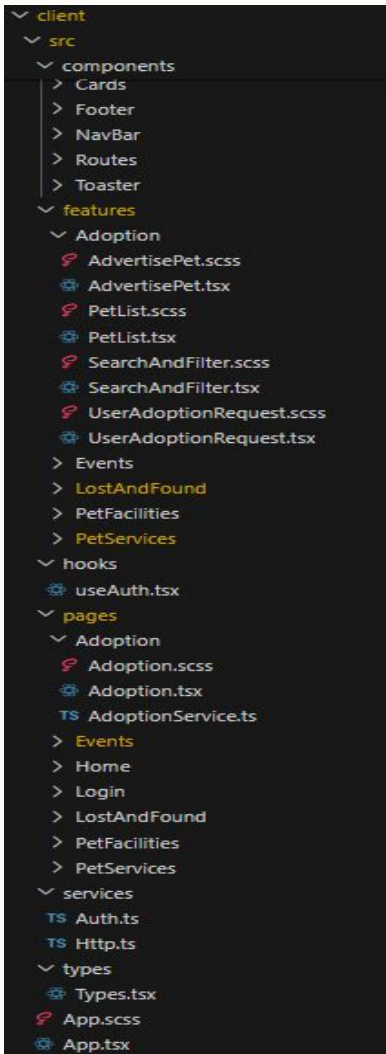




# React application folder structure

# 03





“**components**” - Holds reusable components that are shared across multiple features.

“**features**” - Each feature has its own folder, which contains all its tightly coupled components, hooks, services, and styles. This encapsulation helps in managing feature-specific logic and makes the codebase more modular.

“**hooks**” - Contains reusable custom hooks that are not specific to any single feature.

“**pages**” - Contains page-level components. Each page typically corresponds to a route in your application and aggregates multiple components to form a complete view.

“**services**” - Includes services that are shared between many features.

“**types**” - Includes interfaces.



# Nodejs Folder Structure

# 03





```
✓ SAD-01-24-PETPAL
  > client
  ✓ server
    ✓ config
      JS database.js
      JS helper.js
    ✓ controllers
      JS adoptionController.js
      JS authController.js
      JS careTakersBookingController.js
      JS careTakersController.js
      JS dayCaresBookingController.js
      JS dayCaresController.js
      JS eventController.js
      JS lostAndFoundPetsController.js
      JS petController.js
      JS petServiceCentersController.js
    ✓ models
      JS adoptionRequestModel.js
      JS careTakersBooking.js
      JS careTakersModel.js
      JS dayCareModel.js
      JS dayCaresBooking.js
      JS eventsModel.js
      JS lostAndFoundPetsModel.js
      JS petServiceAppointment.js
      JS petServiceCentersModel.js
      JS petsModel.js
    > node_modules
    ✓ routes
      JS adoptionRoutes.js
      JS authRoutes.js
      JS careTakersBookingRouter.js
      JS careTakersRoutes.js
      JS dayCareRoutes.js
      JS dayCaresBookingRoutes.js
      JS eventRoutes.js
      JS lostAndFoundPetsRoutes.js
      JS petRoutes.js
      JS petServiceCentersRoutes.js
    ✓ utils
      JS constants.js
```

**“config”** - This folder holds configuration files and settings for the server.

**“controller”** - The files in this folder process requests, interact with models, and send responses back to the client. They act as an intermediary between the routes and models.

**“models”** - The files in this folder define the structure of the data, the relationships between different data entities, and include functions for interacting with the database (e.g., schemas for MongoDB using Mongoose).

**“routes”** - Contains handlers that map HTTP requests to specific controller functions, defining how different endpoints of API should respond.

**“utils”** - This folder contains reusable pieces of code that can be used across various parts of application to perform common tasks.

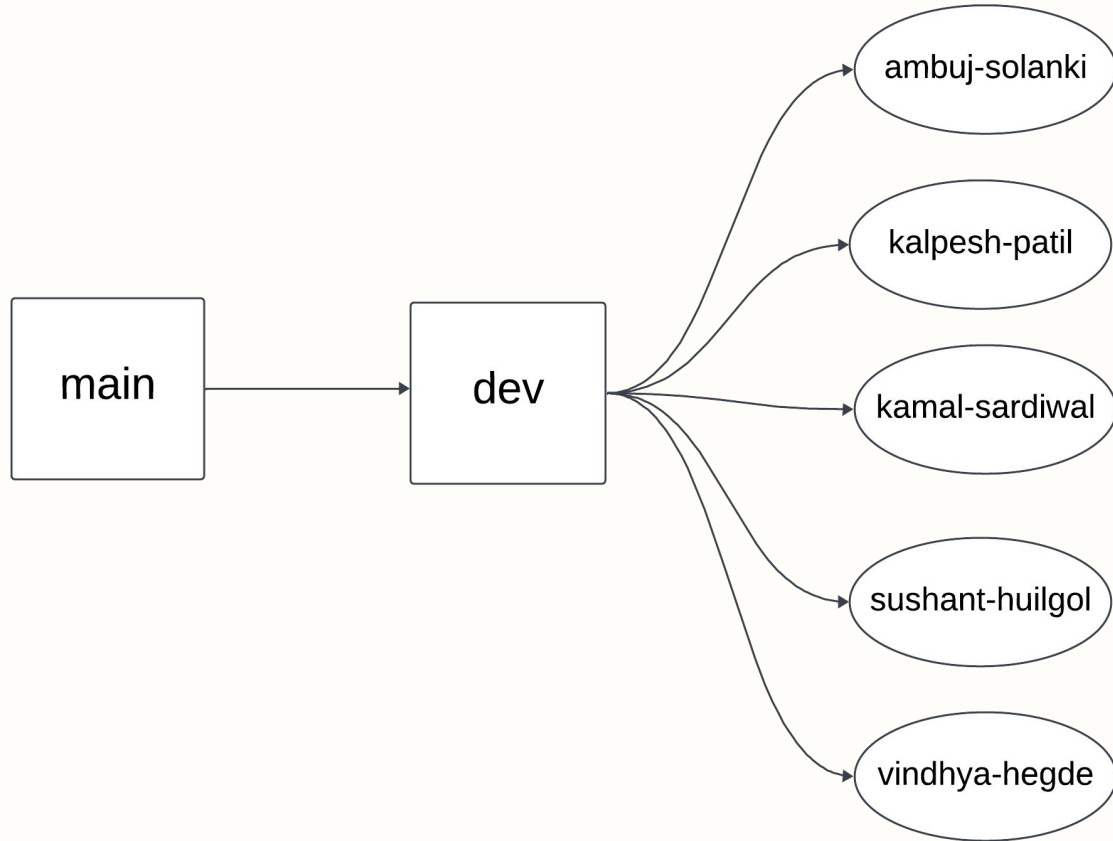


# Git Structure (sad-01-24-petpal)

# 04



# Branch Structure



# Branch Rules

## **main & dev branch**

- Require a pull request before merging
- Require approvals from 2 approvers
- Do not allow bypassing the above settings
- Restrict who can push to matching branches

## **feature branches**

- Restrict who can push to matching branches
- Restrict pushes that create matching branches

# Commit Message Prefixes Followed

- **feat:** A new feature
- **fix:** A bug fix
- **docs:** Documentation only changes
- **style:** Changes that do not affect the meaning of the code (white-space, formatting, missing semi-colons, etc)
- **refactor:** A code change that neither fixes a bug nor adds a feature
- **perf:** A code change that improves performance
- **test:** Adding missing tests or correcting existing tests
- **chore:** Changes to the build process or auxiliary tools and libraries such as documentation generation

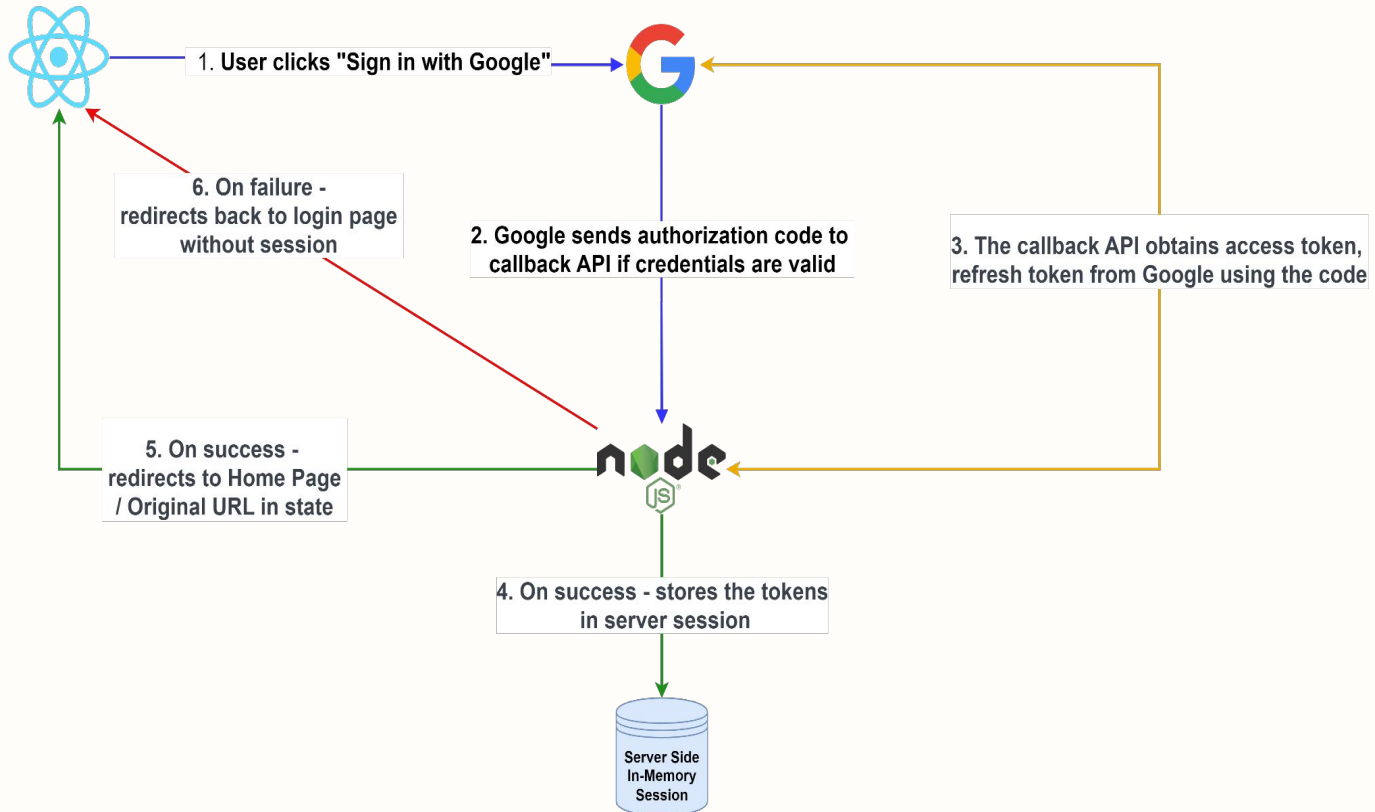


# Google OAuth Login

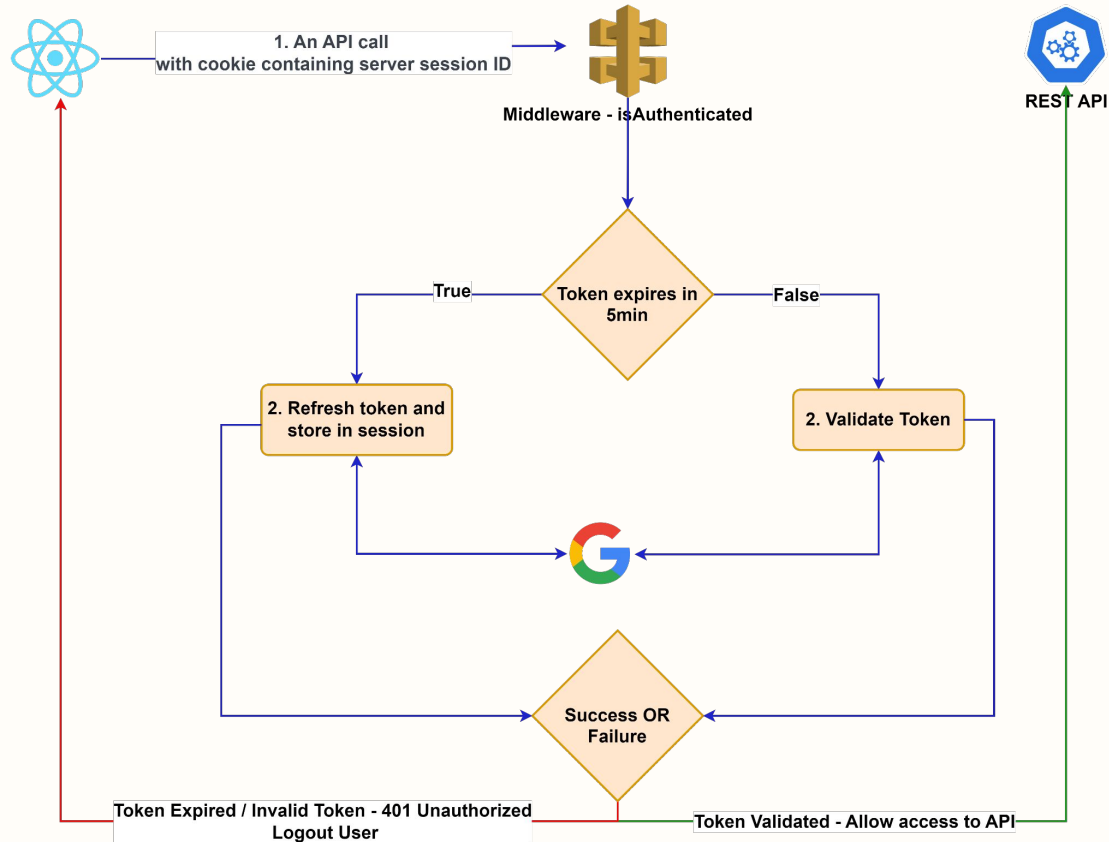
# 05



# OAuth2 Login Flow



# Authenticated User Flow - Token Validation





The background is a light cream color with several large, soft-edged abstract shapes in light blue, yellow, and pink. Scattered around are various small icons: a cluster of red dots in the top left, a red target symbol on the right, a cluster of yellow dashes in the bottom right, and a green plant-like icon at the bottom right.

# Demo



**Thanks for your  
attention.**

**Danke für Ihre  
Aufmerksamkeit.**

