PROJECT DETAILS FORM: dev.rdolcegroup.com

OVERVIEW

MERN INSTALLATION	Primary Domains	Location Paths
Client: React Server: MongoDB, Express, Node.js	dev.rdolcegroup.com	$NAS\web_share\ rdolcegroup.com
	http://rdolcegroup.localhost	\Users\mrsdo\Sites\dreamhost\rdolcegroup\apps.rdolcegroup.com
	GitHub	https://github.com/mrsdo/apps.rdolcegroup.com

CONTACTS

Client or Sponsor	Budget	Team Communications Lead
R. Dolce Group Pine Timber Ridge Office Fort Myers, FL, 33913	$xxxx.xxx	

REQUIREMENTS

Milestones

- Setup Local/Remote application structure, documentation, workflows, and processes (GIT/DEV)
- Create an express-server environment with globals to manage imported packages
- Research potential production environments [Vercel, Netlify, Gatsby Cloud] - Ongoing

TEAM

Lead Programmer/Developer	Team Members & Stakeholders
Martinique Dolce xxx-xxx-9482 marti.dolce@29signals.org	

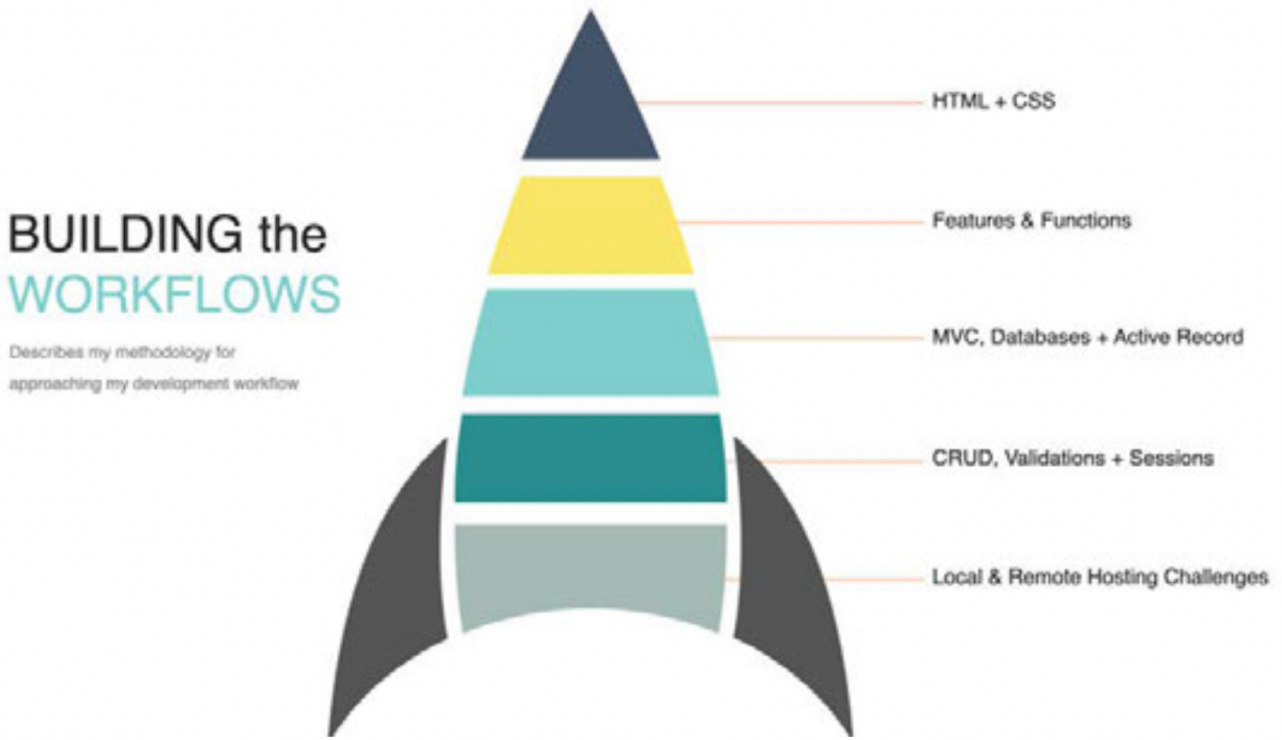
RESEARCH & PLANNING

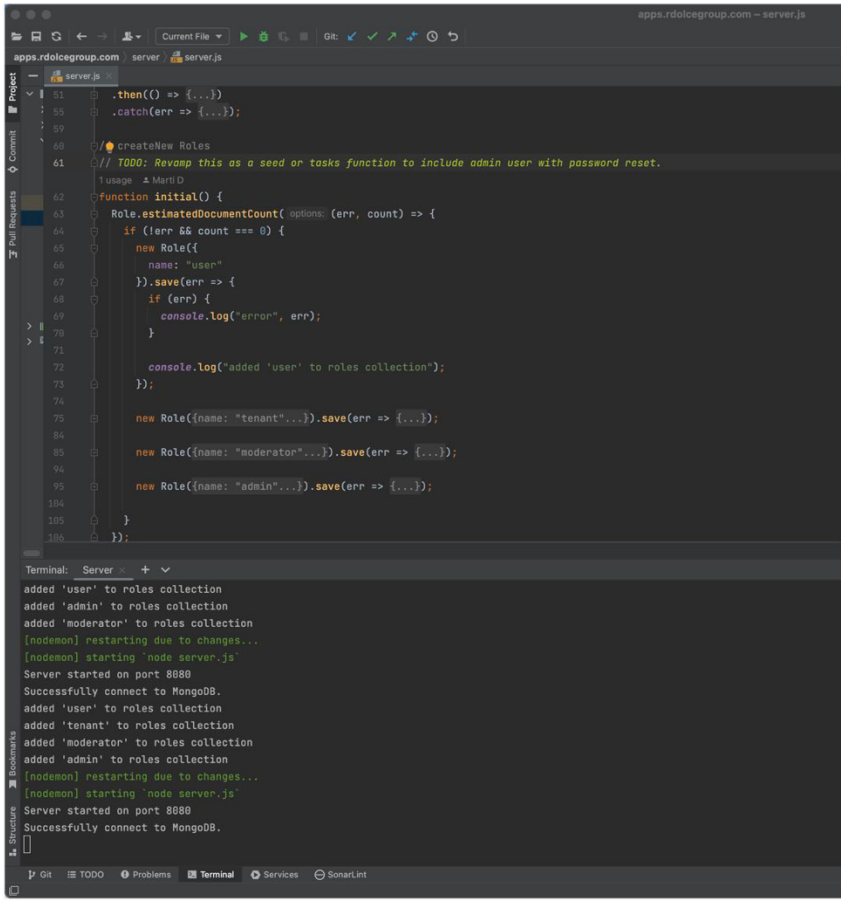
WORKFLOWS

This section explores a bottom-up approach to the development of this application. Starting with the environment or production platform, its limitations, and how those limitations may impact the site's overall functionality and outcomes.

Core specifications require:

- Property Listings
- Users
- Roles
- MongoDB Atlas with Express, React and Node.js
- Private Routing requires an administrative role with authentication





Managing Permissions & Access Via Roles

The question of how to add required data on server launch may require the concept of seeding, similar to ruby. I'll need to do more research on seeding data using Mongoose and possibly JSON.

Maybe something like:
https://stackoverflow.com/questions/44427563/what-is-the-best-way-to-seed-data-in-nodejs-mongoose

https://github.com/mrsdo/react-user-onboarding/blob/main/server/app/models/role.model.js

Authentication with Private Routes

Preventing users from editing core collections will require implementing roles. In this case, the roles are Admin, Moderator, and user.

Admin: GET/POST/PUT/EDIT/DELETE – ALL
Moderators: GET/POST/PUT/EDIT – Assigned Listing | Personal Account
Users: GET – Listings | EDIT – Personal Account

Adding Roles at Server Launch

Server.js adds roles to MongoDB on launch.

DATA: Roles are automatically added to MongoDB via Mongoose				
FIELDS	DATATYPE	NULL (Y/N)	KEY	EXTRA
RoleID	INT	N	PRI	Auto_Inc
RoleName	STR	N	N	-

ADMIN ROUTING: Roles are limited to administrative login			
VERB	ROUTE	ACTION: RolesRouter ('/')	DESCRIPTION
GET	'/roles'	@role = Role.All	.all((req, res, next) or .get((req, res)
GET	'/roles/new'	@role = Role.new via form	.post((req, res) + ${req.body.RoleName}
POST	'/roles'	@role = Role.put(role_params) + redirect	Registration_success.js
GET / ID	'/roles/:id'	@role = Role.find(params[:id])	Find a specific Role
GET /EDIT ID	'/roles/:id/edit'	@role = Role.update(params[:id]) + flash	Edit a specific Role
PUT/PATCH	'/roles/:id'	@role.update(role_params) + flash	.put(async + promise)
DELETE	'/roles/:id'	@role.destroy(params[:id])	.delete((req, res, next) + role.id