Here’s a suggested **project structure** for your RealForGood real estate portal. This structure is modular, scalable, and designed to support both agents and buyers efficiently.

**🏗️ RealForGood Project Structure**

**1. Frontend**

Framework: **React.js** (or **Flutter** for mobile)

/frontend

│

├── /public # Static assets

├── /src

│ ├── /components # Reusable UI components (buttons, cards, forms)

│ ├── /pages # Page views (Home, Listings, Profile, Dashboard)

│ ├── /hooks # Custom React hooks

│ ├── /contexts # Global state (auth, user, listings)

│ ├── /services # API calls

│ ├── /utils # Helper functions

│ └── App.js # Main app component

└── index.js # Entry point

**2. Backend**

Framework: **Node.js + Express** (or **Django REST Framework**)

/backend

│

├── /controllers # Logic for handling requests

├── /models # Database schemas (User, Listing, Inquiry)

├── /routes # API endpoints

├── /middleware # Auth, error handling

├── /services # Business logic (notifications, payments)

├── /config # DB and environment configs

└── server.js # Entry point

**3. Database**

Recommended: **PostgreSQL** or **Firebase Firestore**

* Tables/Collections:
  + Users (agents, buyers)
  + Listings (properties, land)
  + Favorites
  + Messages
  + Payments

**4. Mobile App (Optional)**

Framework: **Flutter** or **React Native**

/mobile

│

├── /screens # Home, Search, Listing Detail, Profile

├── /components # UI elements

├── /services # API integration

├── /utils # Helpers

└── main.dart # Entry point (Flutter)

**5. Admin Panel**

For managing users, listings, and analytics

/admin

│

├── /dashboard # Overview of activity

├── /users # Manage agents and buyers

├── /listings # Approve/edit/delete listings

├── /reports # Analytics and insights

└── /settings # Platform configuration

**6. DevOps & Deployment**

* **Version Control**: GitHub
* **CI/CD**: GitHub Actions or Jenkins
* **Hosting**: AWS, Google Cloud, or Vercel
* **Monitoring**: Sentry, LogRocket
* **Containerization**: Docker (optional)

Would you like me to generate a **folder structure template** or a **GitHub repository starter pack** for this project? I can also help with setting up your first API or frontend page.

✅ Your **RealForGood project structure** and **GitHub starter pack** are ready!

**📁 Folder Structure Includes:**

* **Frontend** (React)
* **Backend** (Node.js)
* **Mobile** (Flutter)
* **Admin Panel**
* **DevOps Setup**

**📦 Starter Pack Files:**

* [Download .gitignore](blob:https://outlook.office.com/b4dfe0ca-502d-4bc5-8431-2ce35785558c) – Preconfigured for Node, Python, Flutter, and common tools.
* [Download README.md](blob:https://outlook.office.com/33199ff7-086c-45ee-8ece-051d54bbf605) – Project overview for GitHub.

**🔌 Next Step: First API Endpoint Setup**

Let’s create a basic **Node.js Express API endpoint** for agents to add property listings.

**🧱 Example: POST /api/listings**

// backend/routes/listings.js

const express = require('express');

const router = express.Router();

// Sample POST endpoint

router.post('/', (req, res) => {

  const { title, price, location, description } = req.body;

  // Save to database (mocked here)

  res.status(201).json({ message: 'Listing created', data: { title, price, location, description } });

});

module.exports = router;

// backend/server.js

const express = require('express');

const app = express();

const listingsRoutes = require('./routes/listings');

app.use(express.json());

app.use('/api/listings', listingsRoutes);

app.listen(3000, () => {

  console.log('Server running on http://localhost:3000');

});

Would you like me to generate this code into files for you, or help you test it locally or on Replit?

✅ Your **API endpoint code files** for RealForGood are ready and include:

**🔐 Authentication Middleware**

* Protects the POST /api/listings route using a simple token (Bearer mysecrettoken).

**📁 Backend Folder Structure**

* backend/server.js – Main server file
* backend/routes/listings.js – API route for adding listings
* backend/middleware/auth.js – Auth middleware

**📦 Starter Files**

* [.gitignore](blob:https://outlook.office.com/c1b52334-3549-44da-a7dd-ccaaa18e7c36)
* [README.md](blob:https://outlook.office.com/afc51a84-61d4-4978-8f8d-2e46beef6a10)

**🧪 How to Test Locally**

**1. Install Dependencies**

npm init -y

npm install express

**2. Run the Server**

node backend/server.js

**3. Test the Endpoint**

Use **Postman** or **cURL**:

curl -X POST http://localhost:3000/api/listings \

  -H "Authorization: Bearer mysecrettoken" \

  -H "Content-Type: application/json" \

**🚀 Deploy on Replit**

1. Go to Replit
2. Create a new Node.js project
3. Upload the backend folder files
4. Run server.js and test using Replit’s web preview or Postman

Would you like me to help you deploy this to Replit now or add more endpoints (e.g., GET /listings, DELETE /listing/:id, or user registration)?