Comprehensive Real Estate Insights Platform

Members:

Donovan Murphy - dmurphy2021@my.fit.edu

Jonathan Bailey - jbailey2021@my.fit.edu

Enrique Obregon - eobregon2020@my.fit.edu

Faculty Advisor & Client:

Fitzroy Nembhard - fnembhard@fit.edu

- Florida Institute of Technology, Department of Computer Science -

User & Developer Guide

Table Of Contents

User Guide

---

1. Introduction

1.1 Purpose of the Manual

This manual provides a comprehensive guide for users of **RealEase**, a real estate insights platform. It explains how to install, navigate, and utilize the application's primary features. This manual is intended to help homebuyers, real estate investors, and general users make informed decisions using the tools provided by RealEase.

1.2 Overview of the Application

RealEase is a full-stack web-based platform that delivers in-depth real estate market intelligence through four main features:

- **Property Listings**: Search for homes that are currently for sale or recently sold in any U.S. ZIP code.
- **Home Comparison Tool**: Compare up to three homes side-by-side with key metrics.
- **ROI Calculator**: Estimate financial return on selected properties using historical and forecasted data.
- **Market Insights Dashboard**: View city-level home value trends using Zillow's ZHVI data.

By consolidating powerful real estate features into one seamless UI, RealEase offers users a streamlined, data-centric homebuying and investment experience.

1.3 Document Conventions

- **Bold** text indicates UI components such as buttons or menu items.
- `Monospace` indicates code or user input.
- Numbered and bulleted lists outline tasks or details.

2. Getting Started

2.1 System Requirements

- **Browser**: Chrome, Firefox, Edge, or Safari
- **Device**: Laptop or desktop with minimum 1366x768 resolution
- **Internet**: Required for accessing real-time data APIs

2.2 Installation/Setup Instructions

RealEase is a fully web-based application with no installation needed.

1. Navigate to the application's deployment URL.
2. Allow permission for location if prompted (optional for nearby listings).
3. Ensure JavaScript is enabled in your browser.

2.3 Account Setup & Login

1. Click **Sign Up** from the homepage.
2. Fill out the registration form with name, email, and password.
3. Submit the form and confirm your email via a 6-digit verification code.
4. Log in using your credentials to access full platform functionality.

3. Using the Features of the System

3.1 Feature Overview

- Search and display homes by ZIP code
- Compare up to three homes
- Calculate estimated ROI on any property
- View housing market trends via ZHVI

3.2 Detailed Feature Walkthroughs

3.2.1 Feature 1: Search Homes by ZIP Code

- Input ZIP code into the **Search** bar.
- View results in two sections:
    - **Recently Sold** homes
    - **Currently For Sale** homes
- Each card displays:
    - Price, bed/bath, square footage, sale date/listing date
    - Mini map view

3.2.2 Feature 2: Home Comparison Tool

- Select up to three listings using the **Compare** checkboxes.
- Click **Compare Homes** to view all side-by-side.
- Comparison grid includes:
    - Price per square foot
    - Age of property
    - HOA fees (if applicable)
    - School scores
    - Walkability score
    - Estimated ROI (if available)

3.2.3 Feature 3: ROI Calculator

- Select any home and click **View ROI**.
- Enter parameters:
    - Down payment
    - Expected rent
    - Loan interest rate
    - Investment timeline
- Results include:
    - Annual cash flow
    - Net ROI after expenses
    - Visualization of value over time

3.2.4 Feature 4: Market Insights Dashboard

- Navigate to the **Market Dashboard**.
- Search by city name (e.g., “Orlando, FL”).
- View:
    - ZHVI historical trend chart
    - YOY percentage change
    - Heat map of recent growth
    - Commentary from latest Zillow reports (if integrated)

3.3 Navigation and Interface Tips

- Use the **Top Navigation Bar** to switch between modules: Home Search, Compare, ROI Tool, Dashboard.
- Hover over icons for tooltips.
- Mobile layout supported but optimized for desktop.

4. User-Specific Examples

4.1 End Users vs. Administrators

- **End Users**: Search listings, compare homes, and calculate ROI.
- **Administrators**: Manage data syncing, maintain APIs, update pricing rules.

4.2 Role-Based Scenarios

4.2.1 Example 1: A First-Time Homebuyer

Maria wants to buy her first home in Denver. She:

- Searches "80202"
- Compares three homes within budget
- Uses ROI tool to evaluate long-term rental potential
- Reviews ZHVI data to confirm Denver’s growth trend

4.2.2 Example 2: A Real Estate Investor

Jayden is scouting for properties in Tampa, FL. He:

- Searches "33612"
- Selects condos under $250k
- Compares ROI for each, based on projected rental yield
- Uses the Market Dashboard to confirm strong 5-year appreciation

5. Troubleshooting and FAQs

5.1 Common Issues

- **Search not loading**: Ensure ZIP is valid and the internet is stable.
- **No ROI data**: Some homes may lack historical pricing or rent data.
- **Login issues**: Try password reset or clear browser cache.

5.2 FAQ Section

- **Q: Can I export comparisons?**
  A: Yes. Use the **Export to PDF** button in the comparison screen.
- **Q: How often is listing data updated?**
  A: Listing and sales data is synced every 24 hours.
- **Q: What sources are used for data?**
  A: Zillow, Redfin, Realtor.com, SchoolDigger, and others via public APIs.
- **Q: Can I save my favorite homes?**
  A: Yes. Logged-in users can bookmark homes for future reference.

6. Support and Contact Information

6.1 Customer Support

For questions or issues:

- Email: dmurphy2021@my.fit.edu

6.2 Feedback Mechanisms

Submit suggestions via the **Feedback Form** in your account settings.

Developer Guide

---

1. Introduction

1.1 Purpose of the Developer Manual

This manual is intended to guide software engineers and contributors in understanding, maintaining, and expanding the RealEase application. It details the system architecture, code structure, API endpoints, integration methods, and best practices.

1.2 Audience

- Full-stack developers
- Frontend or backend specialists
- QA engineers and testers
- DevOps engineers involved in CI/CD and deployment

1.3 Document History and Versioning

- **Version 1.0 – April 2025**: Initial release
- **Version 1.1 – Planned**: Integration of predictive analytics module

2. System Architecture Overview

2.1 Architecture Diagram

[Frontend (React.js)] ⇆ [Backend (Node.js + Express)] ⇆ [MongoDB]

↓

[External APIs]

2.2 Component Descriptions

2.2.1 Frontend

- Built using **React.js** with **Redux** for global state
- Styled using **Tailwind CSS**
- Responsive design with support for mobile and tablet

2.2.2 Backend

- **Express.js** REST API
- Controllers handle logic for each feature: Search, Compare, ROI, Dashboard

2.2.3 Integration

- APIs accessed via **Axios** from the frontend
- External APIs used:
    - Zillow/Redfin (for listings and ZHVI)
    - HomeHarvest (Housing data)

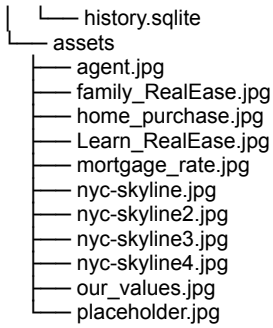
2.3 Data Flow and Interactions

1. User makes request in UI
2. Request hits Express endpoint
3. Backend either queries MongoDB or external API
4. Backend returns formatted JSON response
5. Frontend renders the data

3. Source Code Structure

3.1 Directory Layout

```
WebFrontend
├── .env
├── .gitignore
├── README.md
├── app.py
├── check_mongodb.py
├── config.py
├── process_data.py
├── requirements.txt
├── script.js
├── .git
│   ├── COMMIT_EDITMSG
│   ├── config
│   ├── description
│   ├── FETCH_HEAD
│   ├── HEAD
│   ├── index
│   ├── ORIG_HEAD
│   ├── packed-refs
│   ├── hooks
│   │   ├── applypatch-msg.sample
│   │   ├── commit-msg.sample
│   │   ├── fsmonitor-watchman.sample
│   │   ├── post-update.sample
│   │   ├── pre-applypatch.sample
│   │   ├── pre-commit.sample
│   │   ├── pre-merge-commit.sample
│   │   ├── pre-push.sample
│   │   ├── pre-rebase.sample
│   │   ├── pre-receive.sample
│   │   ├── prepare-commit-msg.sample
│   │   ├── push-to-checkout.sample
│   │   ├── sendemail-validate.sample
│   │   └── update.sample
│   ├── info
│   │   ├── exclude
│   │   └── refs
│   ├── logs
│   │   ├── HEAD
│   │   └── refs
│   │       ├── heads
│   │       │   ├── Donovan
│   │       │   └── main
│   │       └── remotes
│   │           └── origin
│   │               ├── Donovan
│   │               └── main
│   ├── objects
│   │   ├── info
│   │   │   ├── commit-graph
│   │   │   └── packs
│   │   ├── pack
│   │   │   ├── pack-*.idx
│   │   │   ├── pack-*.pack
│   │   │   └── pack-*.rev
│   │   └── [hash folders] (e.g., 13, 23, 2a, ...)
│   └── refs
│       ├── heads
│       │   └── main
│       └── remotes
│           └── origin
│               ├── Donovan
│               └── main
├── .qodo
```



3.2 Module and File Descriptions

3.2.1 Core Files

- `client/src/App.js`: Frontend routing
- `server/index.js`: API entry point

3.2.2 Configuration Files

- `.env`: Environment variables for secrets, ports, and API keys
- `config/db.js`: MongoDB connection handler

3.3 Environment Setup

1. Clone the repository
2. Create `.env` files for frontend and backend
3. Run `npm install` in both `client/` and `server/`
4. Start backend with `npm run dev` and frontend with `npm start`

4. Code Conventions and Guidelines

4.1 Coding Standards

- JavaScript (ES6+)
- Consistent linting via ESLint
- Formatting via Prettier

4.2 Version Control and Branching Strategy

- GitHub for version control
- Branch structure:
    - `main`: production-ready code
    - `dev`: ongoing development
    - `feature/*`: individual feature development

4.3 Code Commenting and Documentation

- Use JSDoc for all functions
- Inline comments for complex logic
- README in each module directory

5. Detailed Documentation of Methods and Functions

5.1 API Documentation

GET /api/homes/:zip

→ Returns listings for a ZIP code

GET /api/compare?h1=:id1&h2=:id2&h3=:id3

→ Returns side-by-side data

POST /api/roi

→ Accepts input params, returns ROI calculation

GET /api/market/:city

→ Returns ZHVI and trend data

5.2 Class and Function Descriptions

5.2.1 Major Functions

- `calculateROI(input)`
    - Inputs: price, rent, expenses, appreciation
    - Returns: annual yield, 5-year ROI, cash flow breakdown
- `compareHomes(ids[])`
    - Inputs: array of Mongo IDs
    - Returns: normalized data objects for UI rendering

5.2.2 Methods/Functions

- `fetchZHVI(city)` – Retrieves Zillow Home Value Index
- `formatCurrency(val)` – Formats numbers to USD
- `normalizeListingData(raw)` – Harmonizes third-party data to internal schema

5.3 Utility and Helper Functions

- `validateZip(zip)` – Ensures ZIP format
- `estimateAppreciation(zip)` – Predicts future prices
- `logError(err)` – Universal error logging

6. Development and Debugging Practices

6.1 How to Add New Features

1. Create a `feature/new-feature-name` branch
2. Add logic in appropriate route/controller
3. Test API locally with Postman
4. Connect frontend component
5. Submit pull request to `dev` for review

6.2 Maintenance Guidelines

- Monitor third-party API limits and keys
- Backup MongoDB monthly
- Archive old listing data quarterly

6.3 Debugging Procedures

- Use Postman to isolate backend bugs
- Use Redux dev tools and Chrome DevTools for frontend
- Implement logging in backend using `winston`

6.4 Testing

- Frontend: Manual and Jest component tests
- Backend: Jest and Supertest for routes/controllers

7. Build, Deployment, and CI/CD Processes

7.1 Build Process

- Frontend: `npm run build` → generates production React files

7.2 Deployment Instructions

- Frontend hosted on Vercel
- Backend on Render or AWS EC2
- Environment variables managed via deployment dashboards

7.3 Continuous Integration/Continuous Deployment (CI/CD)

- GitHub Actions runs tests on PRs
- Merge to `main` triggers deployment webhook

---

8. Appendices and Additional Resources

8.1 Glossary of Terms

- **ZHVI**: Zillow Home Value Index
- **ROI**: Return on Investment
- **API**: Application Programming Interface
- **CRUD**: Create, Read, Update, Delete

8.2 External Resources

- React Documentation
- Express.js Docs
- MongoDB Manual
- Zillow ZHVI Guide