



**Dr. AMBEDKAR INSTITUTE OF TECHNOLOGY**

Near Jnana Bharathi Campus, Bengaluru-560 056.

(An Autonomous Institution, Aided by Government of Karnataka)

**Department of computer science & engineering 2023-24**

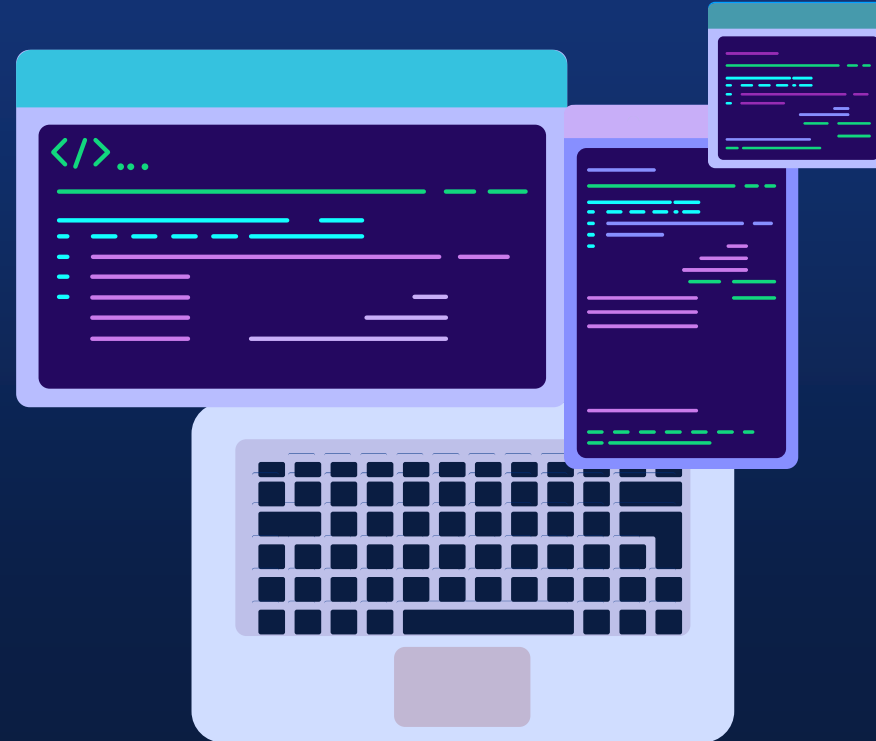
# DBMS MINI PROJECT

Course Code : 21CLS505

Mini project on

“Streamlining Room Management and Rentals”

Bachelor of engineering in  
**Computer science and engineering**



# Streamlining Room Management and Rentals:

## PROJECT MEMBERS:

1DA21CS092: NADDANA YASHWANTH

### BACKEND:

Design Of Schema.

- Listing Schema.
- Review Schema.
- User Schema .
- With Schema and Entity Rational Diagrams.

1DA22CS409: HAJARATALI S MOGALLAI

### FRONTEND:

- Index page
- Show page
- Commenting
- Review
- With Express js

### BACKEND:

- Writing Routes
- Writing Methods
- With node js



## **Project Overview:**

The project seems to be a property rental and listing platform where users can browse, create, and manage property listings. It includes features for creating, updating, and deleting property listings, managing reviews, categorizing listings by type, and implementing a search functionality. Additionally, there are modules for user authentication, menu management, and the creation of rental menus associated with specific listings.

## **Key Features and Functionalities:**

### **1. Property Listings:**

- Users can view a list of all property listings on the homepage.
- Detailed information about each listing, including images, is displayed on individual listing pages.
- The project allows for the creation, editing, and deletion of property listings.

### **2. User Authentication:**

- Users can sign up for accounts, log in, and log out.
- Authentication is integrated with Passport.js, providing secure user registration and login processes.

### **3. Menu Management:**

- There is a module for creating and managing menus associated with specific listings.
- Menus include information about different items, and users can create, edit, and delete menu items.

### **4. Search Functionality:**

- The project includes a search feature allowing users to search for listings based on titles, categories, or locations.

### **5. Listing Categories:**

- Property listings are categorized based on types such as Cottage, Residential, Commercial, Vacation, Apartment, Condo, Townhouse, and Other.





## **6.Review System:**

- Users can leave reviews for specific property listings.
- The reviews are associated with the respective listings and are displayed on the listing detail pages.

## **7.User Profile:**

- Users have profiles where they can manage their information and potentially view their rental history.

## **8.Trending Listings:**

- There is a module to display trending listings based on reviews with a rating of 5.

## **9.Search Functionality:**

- Users can perform searches for listings based on titles, categories, and locations.

## **10.Error Handling and Flash Messages:**

- The project includes error handling to manage potential issues during user interactions.
- Flash messages are used to provide feedback to users after certain operations.

## **Additional Modules:**

### **•Menu Integration:**

- There is a separate set of modules related to menu creation and management, allowing users to add menus to specific listings.

### **•User Authentication and Authorization:**


- Passport.js is used for user authentication, providing a secure and customizable authentication system.

### **•Rental Module:**

- There is a module related to renting properties, allowing users to book and manage rental periods.
- 



## Considerations:

- The project appears to be built using a Node.js framework, possibly Express.js, and is likely connected to a MongoDB database.
  - Middleware like **connect-flash** is used for displaying flash messages to users.
  - File uploads, possibly for listing images, are handled using middleware that works with file paths and filenames.
- 

# Streamlining Room Management and Rentals: BACKEND: Design Of Schema.



User:

User_id	User_name	User_email	User_password	Rents_Id	Menu_Id
---------	-----------	------------	---------------	----------	---------

Listing:

Listing_Id	Title	Description	Image	Category	Price	City	Location	Reviews_id	Owner_id
------------	-------	-------------	-------	----------	-------	------	----------	------------	----------

Reviews:

Reviews_id	Comments	Rating	Createdat	Author_id	Listing_Id
------------	----------	--------	-----------	-----------	------------

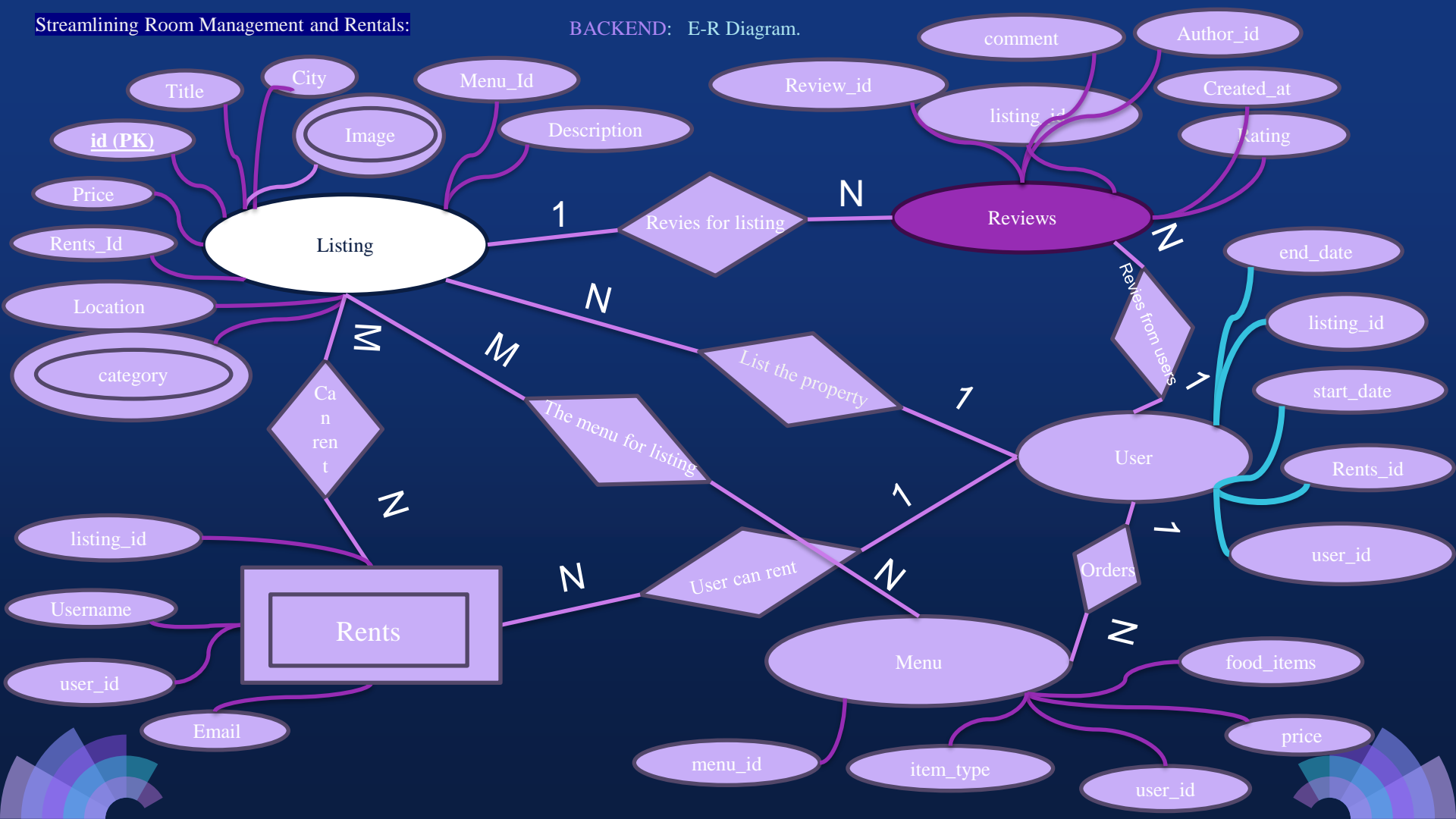
Rent:

Rent_id	SatartDate	EndDate	Listing_Id
---------	------------	---------	------------

Menu:

Menu_id	Item_type	Food_item	price	Listing_id
---------	-----------	-----------	-------	------------





## User Schema:

```
const mongoose = require('mongoose');

const userSchema = new
mongoose.Schema({
  username: String,
  email: String,
  password: String
});

const User = mongoose.model('User',
userSchema);

module.exports = User;
```

## Review Schema:

```
const mongoose = require('mongoose');

const reviewSchema = new mongoose.Schema({
  property_id: mongoose.Schema.Types.ObjectId,
  user_id: mongoose.Schema.Types.ObjectId,
  rating: Number,
  comment: String
});

const Review = mongoose.model('Review',
reviewSchema);

module.exports = Review;
```





## listing Schema

```
const mongoose = require('mongoose');

const propertySchema = new mongoose.Schema({
  title: String,
  description: String,
  image: { url: String, filename: String },
  price: Number,
  location: String,
  country: String,
  category: String,
  owner_id: mongoose.Schema.Types.ObjectId,
  reviews: [{ type:
mongoose.Schema.Types.ObjectId, ref: 'Review' }],
  __v: Number
});

const Property = mongoose.model('Property',
propertySchema);
module.exports = Property;
```

## Rent Schema

```
const mongoose = require('mongoose');

const rentSchema = new mongoose.Schema({
  property_id: mongoose.Schema.Types.ObjectId,
  start_date: Date,
  end_date: Date
});

const Rent = mongoose.model('Rent', rentSchema);

module.exports = Rent;
```

## Menu Schema:

```
const mongoose = require('mongoose');

const menuSchema = new mongoose.Schema({
  property_id:
mongoose.Schema.Types.ObjectId,
  dish_name: String
});

const Menu = mongoose.model('Menu',
menuSchema);

module.exports = Menu;
```

```
const listings = await listing.find({});
```

id	title	description	image_url	price	location	City
1	Rose Cottage.	Cozy cottage wi	<a href="https://example.com">https://example.com</a>	12000	Bangalore	Bangalore
2	Silver Springs Towers	Luxury resident	<a href="https://example.com">https://example.com</a>	25000	Whitefield	Bangalore
3	Harmony Gardens	Tranquil residen	<a href="https://example.com">https://example.com</a>	30000	Jayanagar	Bangalore
4	Kanakapura Cottage	Idyllic cottage in	<a href="https://example.com">https://example.com</a>	13000	Bangalore, Kanakapura	Bangalore
5	Coorg (Kodagu) Cottage	Experience the	<a href="https://example.com">https://example.com</a>	15000	Bangalore, Coorg (Kodagu)	Bangalore
6	Chikmagalur Cottage	Serene hill stati	<a href="https://example.com">https://example.com</a>	1000	Bangalore, Chikmagalur	Bangalore
7	Nandi Hills	Picturesque des	<a href="https://example.com">https://example.com</a>	15000	Bangalore, Nandi Hills	Bangalore

## BACKEND: Backend-Data Quires and Back end Data:

```
const listings = await listing.find({});
```

category	owner	reviews
Cottage	John Doe	[]
Residential	Jane Smith	[65acb575b50176a5024484d5,]
Residential	Robert Johnson	[]
Cottage	Maria Rodriguez	[65acb6cfd35f62331e89ef62]
Cottage	Michael Brown	[65acb6b2d35f62331e89ef54]
Cottage	Emily Davis	[65acb693d35f62331e89ef41]
Vacation	William Johnson	[65acb67cd35f62331e89ef33, 65afe506d62d5af4d685b2e7]

## BACKEND: Backend-Data Quires and Back end Data:

```
const rents = await Rent.find({});
```

id	property_id	start_date	end_date
1	1	01-02-2024	07-02-2024
2	1	15-03-2024	20-03-2024
3	2	10-02-2024	18-02-2024
4	2	01-03-2024	10-03-2024
5	4	25-02-2024	05-03-2024
6	4	10-04-2024	18-04-2024
7	1	01-02-2024	07-02-2024
8	1	15-03-2024	20-03-2024
9	2	10-02-2024	18-02-2024
10	2	01-03-2024	10-03-2024
11	4	25-02-2024	05-03-2024
12	4	10-04-2024	18-04-2024

## BACKEND: Backend-Data Quires and Back end Data:

```
const menuItems = await Menu.find({});
```

id	property_id	dish_name
1	1	Breakfast: Pancakes
2	1	Lunch: Grilled Chicken Salad
3	1	Dinner: Pasta Alfredo
4	2	Breakfast: Avocado Toast
5	2	Lunch: Sushi Bowl
6	2	Dinner: Filet Mignon
7	4	Breakfast: Belgian Waffles
8	4	Lunch: Margherita Pizza
9	4	Dinner: Shrimp Scampi
10	6	Breakfast: Omelette
11	6	Lunch: Caprese Salad
12	6	Dinner: Vegetable Biryani
13	7	Breakfast: French Toast
14	7	Lunch: Chicken Curry
15	7	Dinner: Chocolate Cake
16	1	Breakfast: Pancakes
17	1	Lunch: Grilled Chicken Salad
18	1	Dinner: Pasta Alfredo
19	2	Breakfast: Avocado Toast

## BACKEND: Backend-Data Quires and Back end Data:

```
const reviews = await Review.find({});
```

id	property_id	user_id	rating	comment
1	2	1	5	Great place! Highly recommended.
2	2	2	4	Beautiful view from the towers.
3	2	3	5	Luxurious amenities.
4	2	4	4	Enjoyed the stay at Silver Springs Towers.
5	6	5	3	Nice cottage in Chikmagalur.
6	6	6	4	Peaceful and serene environment.
7	6	7	5	Perfect getaway!
8	7	1	5	Nandi Hills is a fantastic destination.
9	7	2	4	Scenic views and comfortable stay.

```
const users = await User.Find({});
```

id	username	email	password
1	john_doe	john@example.com	hashed_password_1
2	jane_smith	jane@example.com	hashed_password_2
3	robert_johnson	robert@example.com	hashed_password_3
4	maria_rodriguez	maria@example.com	hashed_password_4
5	michael_brown	michael@example.com	hashed_password_5
6	emily_davis	emily@example.com	hashed_password_6
7	william_johnson	william@example.com	hashed_password_7
8	john_doe	john@example.com	hashed_password_1
9	jane_smith	jane@example.com	hashed_password_2
10	robert_johnson	robert@example.com	hashed_password_3
11	maria_rodriguez	maria@example.com	hashed_password_4
12	michael_brown	michael@example.com	hashed_password_5
13	emily_davis	emily@example.com	hashed_password_6
14	william_johnson	william@example.com	hashed_password_7



# Signup for Wanderlust

username:

hazarat

Email:

hajaratali@gmail.com

password:

.....

signup



Explore

Login from

Search Destiny

Search

Airbnb your home

signup

login

# login for Wanderlust

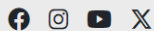
username:

ali

password:


.....

Login





©Wanderlust Private Limited


[Privacy](#)[Terms](#)


 Explore


Main page / Index page


Search Destination  Search [Airbnb your home](#) [Profile](#) [Log out](#)


 Trending


 Cottage


 Residential


 Commercial


 Vacation


 Apartment

 Condo


 Townhouse

 Other


Display total after Taxes 




**Kanakapura cottage**  
Category: Cottage  
₹15000/night +18% GST




**Kanakapura cottage**  
Category: Cottage  
₹13000/night +18% GST




**Coorg (Kodagu) cottage**  
Category: Cottage  
₹15000/night +18% GST



**Coorg (Kodagu) cottage**  
Category: Cottage  
₹15000/night +18% GST



**Coorg (Kodagu) cottage**  
Category: Cottage  
₹15000/night +18% GST



**Coorg (Kodagu) cottage**  
Category: Cottage  
₹15000/night +18% GST

# Main page / Index page

Search Destination

 Search

[Airbnb your home](#)

[Profile](#)

[Log out](#)

Listings searched by Location



 Trending

 Cottage

 Residential

 Commercial

 Vacation

 Apartment

 Condo

 Townhouse

 Other

Display total after Taxes ☐



Yercaud

Category: Vacation

₹8000/night



Nandi Hills

Category: Vacation

₹15000/night



Chikmagalur cottage

Category: Cottage

₹1000/night



## Show Page Details of listing

Search Destiny

Search

Airbnb your home Profile Log out



Owned By: *ali*

Welcome to Urban Oasis Towers, an exclusive retreat in the heart of Bangalore, where modern luxury converges with urban convenience to create a haven for contemporary living. As you enter this architectural masterpiece, prepare to be captivated by a world where every detail is meticulously curated to offer a lifestyle that transcends expectations.

₹32,000

Koramangala

Bangalore

Edit

Delete

Leave a Review

Review From

Rating



Comment

Submit

All Reviews



©Wanderlust Private Limited

[Privacy](#)[Terms](#)

[Explore](#)

# All Reviews

[Search](#)[Airbnb your home](#)[Profile](#)[Log out](#)

## All Reviews

@ali



super good.

[Delete](#)

@ali



I recently had the pleasure of staying at Rose Cottage in Bangalore, and it was truly an amazing experience. The cottage is nestled in a serene location in Bangalore, providing a perfect escape from the hustle and bustle of the city.

[Delete](#)

@Miss kim



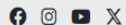
The ambiance of Rose Cottage is charming, with a beautiful blend of modern amenities and a touch of rustic elegance. The interior decor is tastefully done, creating a cozy and comfortable atmosphere. The attention to detail in every aspect of the cottage, from the furnishings to the amenities, exceeded my expectations.

[Delete](#)

@Miss kim



I highly recommend Rose Cottage to anyone looking for a peaceful and rejuvenating getaway in Bangalore. Whether you're a solo traveler, a couple seeking a romantic retreat, or a family looking for a cozy vacation home, Rose Cottage is the perfect choice.

[Delete](#)

©Wanderlust Private Limited

# Edit from

Search Destiny

 Search

Airbnb your home

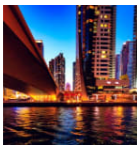
Profile

Log out

Description:

Welcome to Urban Oasis Towers, an exclusive retreat in the heart of Bangalore, where modern luxury converges with urban convenience to create a haven for contemporary living. As you enter this architectural masterpiece, prepare to be captivated by a world where every detail is meticulously curated to offer a lifestyle that transcends expectations.

old image



Upload New Image:

Choose file

No file chosen

Price:

32000

Country:

Bangalore

Location:

Koramangala

Edit



[Explore](#)

# Reviews

[Search](#)[Airbnb your home](#)[Profile](#)[Log out](#)

## All Reviews

@ali



super good.

[Delete](#)

@ali



I recently had the pleasure of staying at Rose Cottage in Bangalore, and it was truly an amazing experience. The cottage is nestled in a serene location in Bangalore, providing a perfect escape from the hustle and bustle of the city.

[Delete](#)

@Miss kim



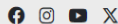
The ambiance of Rose Cottage is charming, with a beautiful blend of modern amenities and a touch of rustic elegance. The interior decor is tastefully done, creating a cozy and comfortable atmosphere. The attention to detail in every aspect of the cottage, from the furnishings to the amenities, exceeded my expectations.

[Delete](#)

@Miss kim



I highly recommend Rose Cottage to anyone looking for a peaceful and rejuvenating getaway in Bangalore. Whether you're a solo traveler, a couple seeking a romantic retreat, or a family looking for a cozy vacation home, Rose Cottage is the perfect choice.

[Delete](#)

©Wanderlust Private Limited

Explore

# Menu-Lunch

Search Destiny

Search

Airbnb your home

Profile

Log out

## Add Menu Item for Urban Oasis Towers

Item Type:

Lunch

Food Items:

- ☒ Caesar Salad - \$8.99
- ☒ Quinoa Bowl - \$10.99
- ☒ Chicken Wrap - \$7.99
- ☒ Vegetarian Pizza - \$12.99
- ☒ Pasta Primavera - \$9.99
- ☒ Caprese Salad - \$7.49
- ☒ Shrimp Tacos - \$11.99
- ☐ Club Sandwich - \$8.99
- ☐ Vegetable Stir-Fry - \$9.49
- ☐ Mushroom Risotto - \$11.49
- ☐ Cobb Salad - \$10.49
- ☐ Chicken Caesar Wrap - \$8.99
- ☐ Tomato Basil Soup - \$6.49
- ☐ Greek Salad - \$7.99
- ☐ Pesto Pasta - \$9.99
- ☐ Tuna Salad Sandwich - \$8.49
- ☐ Chicken Burrito Bowl - \$11.99
- ☐ Spinach and Feta Wrap - \$7.99
- ☐ BBQ Chicken Pizza - \$12.99
- ☐ Veggie Burger - \$9.49

Total Price:

70.43

Order

# Menu-Dinner

Search Destiny

 Search

[Airbnb your home](#)

[Profile](#)

[Log out](#)

## Add Menu Item for Urban Oasis Towers

Item Type:

Dinner

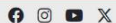
Food Items:

- ☒ Salmon Teriyaki - \$16.99
- ☒ Beef Stroganoff - \$14.99
- ☒ Chicken Alfredo - \$13.99
- ☒ Vegetable Lasagna - \$12.49
- ☐ Seafood Paella - \$18.99
- ☐ Steak Fajitas - \$17.49
- ☐ Lobster Ravioli - \$15.99
- ☐ Vegetable Curry - \$11.99
- ☐ Pork Chops - \$14.49
- ☐ Chicken Marsala - \$13.49
- ☐ Sushi Platter - \$16.99
- ☐ Eggplant Parmesan - \$12.99
- ☐ Shrimp Scampi - \$15.49
- ☐ Beef Tacos - \$13.99
- ☐ Chicken Enchiladas - \$14.99
- ☐ Vegetarian Sushi Roll - \$12.49
- ☐ Chicken Parmesan - \$13.99
- ☐ Grilled Swordfish - \$16.49
- ☐ Vegetable Kebabs - \$11.99
- ☐ Ribeye Steak - \$17.99

Total Price:

58.46

Order



©Wanderlust Private Limited

# Menu-Breakfast

Search Destiny

 Search

[Airbnb your home](#)

[Profile](#)

[Log out](#)

## Add Menu Item for Urban Oasis Towers

Item Type:

Breakfast

Food Items:

- ☒ Pancakes - \$7.99
- ☒ Avocado Toast - \$9.99
- ☐ Pancakes - \$7.99
- ☐ Avocado Toast - \$9.99
- ☒ Eggs Benedict - \$8.49
- ☒ Omelette - \$6.99
- ☒ French Toast - \$7.49
- ☒ Waffles - \$8.99
- ☒ Smoothie Bowl - \$9.49
- ☐ Breakfast Burrito - \$7.99
- ☐ Granola Parfait - \$6.49
- ☐ Bagel with Cream Cheese - \$5.99
- ☐ Fruit Salad - \$5.49
- ☐ Muffins - \$6.99
- ☐ Yogurt Parfait - \$7.49
- ☐ Egg Sandwich - \$6.99
- ☐ Crepes - \$10.99
- ☐ Hash Browns - \$4.99
- ☐ Bacon and Eggs - \$8.99
- ☐ Breakfast Quesadilla - \$7.99
- ☐ Frittata - \$9.49
- ☐ Cinnamon Rolls - \$6.49

Total Price:

59.43

Add Item



# All Quires from All Modules

## **Retrieve all listings:**

```
const All Listings = await Listing.find({ });
```

## **New Form Module:**

Render the form for creating a newlisting:res.render("listing/new.ejs");

## **Show Listing Module:**

Retrieve a specific listing by ID and populate its reviews and owner:

```
const listing = await Listing.findById(id).populate({ path: "reviews", populate: { path: "author" } }).populate("owner");
```

## **Create New Listing Module:**

Create a new listing with the provided data and save it to the database.

## **Edit Listing Module:**

Retrieve a specific listing by ID for editing:const listing = await Listing.findById(id);

## **Update Listing Module:**

Update an existing listing with the provided data and image (if provided).

## **Delete Listing Module:**


Delete a listing by ID :let deleted = await Listing.findByIdAndDelete(id);





To retrieve listings based on different categories

```
// Example for retrieving Cottage listings
const cottageListings = await Cottage(req, res);
// Example for retrieving Residential listings
const residentialListings = await Residential(req, res);
// Example for retrieving Commercial listings
const commercialListings = await Commercial(req, res);
// Example for retrieving Vacation listings
const vacationListings = await Vacation(req, res);
// Example for retrieving Apartment listings
const apartmentListings = await Apartment(req, res);
// Example for retrieving Condo listings
const condoListings = await Condo(req, res);
// Example for retrieving Townhouse listings
const townhouseListings = await Townhouse(req, res);
// Example for retrieving Other listings
const otherListings = await Other(req, res);
```





### getMenuForm:

- Retrieve a specific listing by ID and populate its 'menu' field: `const listingId = req.params.id;`
- `const listing = await Listing.findById(listingId).populate('menu');`

### storeMenu:

- Extract data from the request body: `const listingId = req.params.id;`
- `const { itemType, foodItems, price } = req.body.item;`
- Find the listing with the specified ID: `const listing = await Listing.findById(listingId);`

Create a new **Menu** item:

```
const newMenuItem = new Menu({  
  itemType,  
  foodItems,  
  price,  
});
```

- Push the new menu item to the 'menu' array of the listing:  
    `listing.menu.push(newMenuItem);`
- Save changes to the listing: `await listing.save();`





**Fetch Listing:**Query to fetch a specific listing by its ID.

```
const listing = await Listing.findById(req.params.id);
```

**Handle Listing Not Found:**Check if the listing exists. If not, flash an error message and redirect.

```
if (!listing) { req.flash('error', 'Listing not found'); return res.redirect('/'); }
```

**Create Rent Instance:**Extract start and end dates from the request body.

Create a new **Rent** instance with the extracted dates.

```
const rental = new Rent({ startDate, end Date, });
```

**Save Rental:**Save the rental to the database using **rental.save().await rental.save();**

**Update Listing with Rental:**Ensure the **rents** array is initialized in the listing.

Push the rental into the **rents** array of the listing.

Save the changes to the listing.

```
listing.rents = listing.rents || [];
```

```
listing.rents.push(rental); await listing.save();
```

**Flash Success Message:**

Flash a success message indicating that the listing has been successfully rented.

```
req.flash('success', 'Successfully rented the listing!');
```

**Redirect:**

Redirect to the page displaying details of the rented listing.

```
res.redirect(`/listings/${listing.id}`);
```







## CreateReview:

- Fetch the listing by its ID:

```
let listing = await Listing.findById(req.params.id);
```

- Create a new **Review** instance with the data from the request body:

```
let NewReview = new Review(req.body.review);
```

- Set the author of the review to the current user's ID:

```
NewReview.author = req.user._id;
```

- Push the new review into the **reviews** array of the listing:

```
listing.reviews.push(NewReview);
```

- Save the new review and the updated listing:

```
await NewReview.save(); await listing.save();
```

## DeleteReview:

- Retrieve the listing ID and review ID from the request parameters:

```
let { id, reviewId } = req.params;
```

- Use **findByIdAndUpdate** to pull the specified review ID from the **reviews** array of the listing:

```
await Listing.findByIdAndUpdate(id, { $pull: { reviews: reviewId } });
```

- Use **findByIdAndDelete** to delete the specified review:

```
await Review.findByIdAndDelete(reviewId);
```





## SignupUpForm:

- Purpose: Render the signup form.

```
res.render("../views/users/signup.ejs");
```

## SignUpUser:

- Purpose: Create a new user account.

- Steps:

- Extract username, email, and password from the request body.
- Create a new **User** instance with the provided email and username.
- Use **User.register** method to register the user with the provided password.
- Log in the registered user and redirect to the listings page.

```
let newUser = new User({ email, username }); const registeredUser = await User.register(newUser, password);  
req.login(registeredUser, (err) => { /* ... */ });
```

## userprofile:

- Purpose: Render the user profile page.

```
res.render("users/userprofile.ejs");
```





### LoginForm:

- Purpose: Render the login form.

```
res.render("users/login.ejs");
```

### LoginByUser:

- Purpose: Log in a user


```
req.flash('success', "Welcome Back to wanderlust ");
```

```
let redirectUrl = res.locals.redirectUrl || "/listings"; res.redirect(redirectUrl);
```

### LogOut:

- Purpose: Log out the currently authenticated user.

```
req.logout((err) => { /* ... */ });
```





## **Reservation System**

### **Centralized Reservation System:**

- Implement a centralized reservation system using Property Management Software (PMS).
- Ensure real-time updates to prevent overbooking.
- Enable seamless communication between front-desk operations and online booking platforms.

## **Online Booking**

### **Online Booking Platforms:**

- Provide a user-friendly online booking platform for guests.
- Integrate the online booking system with the centralized reservation system for real-time data synchronization.


## **Check-In and Check-Out Automation**

### **Automated Check-In and Check-Out:**

- Implement automated processes to minimize guest waiting times.
- Explore mobile check-in options for increased guest convenience.

## **Pricing Optimization**

### **Dynamic Pricing Strategies:**

- Implement dynamic pricing based on demand, seasonality, and other factors.
  - Utilize analytics and historical data to optimize pricing decisions.
- 




## Menu Creation and Configuration

### Menu Setup:

1. Define the types of items to be offered in the menu (e.g., appetizers, main courses, beverages).
2. Specify the structure and layout of the menu.

### Item Configuration:

1. Add and configure individual menu items.
  2. Include details such as item name, description, category, and pricing.
  3. Attach images to visually represent each menu item.
- 

A group of four people (three men and one woman) are sitting around a table in a meeting. They are looking at a laptop screen. The image is overlaid with a purple filter. A white rounded rectangle with a thin white border is centered over the image, containing the text 'Thank you' in a bold, white, sans-serif font.

**Thank you**