**Project Design Phase-II**

**Project Development Phase**

| **Date** | **02 November 2023** |
| --- | --- |
| **Team Id** | **NM2023TMID02508** |
| **Project Name** | A CRM Application To Manage The Booking Of Co-Living |

### No. Of Functional Features Included In The Solution

The number of functional features included in a solution for recruiting A CRM Application to manage the booking of co living can vary widely depending on the specific needs of the organization and the complexity of the recruitment process. :

**1.User Management:**

User registration and login.

User profiles with personal details and contact information.

User roles (renters, property owners, administrators)**.**

**2.Property Listings:**

Property listing creation for property owners.

Property details (location, size, amenities, photos).

Pricing and availability management.

**3.Booking Management:**

Property booking system for renters.

Availability calendars.

Payment processing and invoicing.

Booking confirmation and cancellation

**4.Search and Filter Functionality:**

Advanced search and filter options for renters to find co-living spaces.

Location-based search.

Filter by price, amenities, and availability.

**5.Notifications and Alerts:**

Email and in-app notifications for booking updates, messages, and reminders.

Alerts for property owners when a booking request is received.

### Code-Layout, Readability And Re-usability

**Code Layout:**

**1.Consistent Indentation and Formatting**:

Use consistent and clear indentation, such as 4 spaces or tabs.

Follow a consistent code style guide (e.g., PEP 8 for Python).

**2.Descriptive Variable and Function Names**:

Use meaningful and descriptive names for variables, functions, and classes.

Avoid single-letter or cryptic variable names.

**3.Comments and Documentation**:

Include comments to explain complex logic or algorithms.

Document code with clear function and class descriptions

**4.Separation of Concerns**:

Use modular code with well-defined functions and classes.

Follow the Single Responsibility Principle (SRP), ensuring each function or class has one clear

**5.Error Handling**:

Implement proper error handling with meaningful error messages.

Use try-catch or exception handling where appropriate.

**Readability:**

**1.Code Version Control**:

Use a version control system (e.g., Git) to track changes and collaborate with team members

Write unit tests and integration tests to ensure code quality.

Keep tests in a separate directory or module.

**2.Comments and Documentation**:

Include comments to explain complex logic or algorithms.

Document code with clear function and class descriptions.

**Reusability:**

**1.Modular Design**:

Break your code into reusable modules or libraries.

Encapsulate related functionality within classes and packages.

**2.Reusable Functions**:

Identify common functions or utilities that can be reused across different parts of the project

**3.Code Review**:

Conduct code reviews to identify and promote reusable code components.

**Keep Dependencies Managed**:

Use dependency management tools (e.g., package managers) to keep external libraries and packages up-to-date.

By focusing on code layout, readability, and reusability, you can create a recruiting assistance system that is not only functional but also maintainable, adaptable, and user-friendly for HR managers. This approach simplifies ongoing development and maintenance while promoting efficient collaboration among developers.

## Utilization Of Algorithms, Dynamic Programming, Optimal Memory Utilization

**1.Resource Allocation**:

Use dynamic programming to optimize resource allocation, such as managing server resources or database connections efficiently based on demand.

**2.Session Management**:

Employ dynamic programming to manage user sessions efficiently, allowing you to store session data with minimal memory usage.

Search and Recommendation Algorithms:

Apply dynamic programming to optimize search and recommendation algorithms, reducing the computational load when users search for co-living spaces or receive recommendations.

**3.Database Query Optimization**:

Optimize database queries with dynamic programming to ensure that only necessary data is retrieved and stored in memory.

**4.Data Compression**:

Use data compression techniques to reduce the memory footprint when storing large volumes of data, such as images or documents.

**5.Data Structures and Algorithms**:

Choose appropriate data structures and algorithms that minimize memory usage. For example, use efficient data structures like hash maps or balanced trees.

**6.Batch Processing**:

Apply batch processing for resource-intensive tasks to optimize memory utilization and improve performance.

These functional requirements are essential for building a robust data analytics and reporting system that empowers HR managers to analyze data, gain insights, and make informed decisions related to the recruitment process. The specific requirements may vary depending on the organization's size, industry, and unique needs.

### Debugging & Traceability

**1.Logging and Error Handling:**

Implement comprehensive logging to record events, errors, and information about the system's behavior.

Use appropriate logging levels (e.g., debug, info, error) to categorize log entries

**2Debugging Tools:**

Utilize debugging tools and integrated development environments (IDEs) that allow you to set breakpoints, inspect variables, and step through code.

**3.Unit Testing:**

Write unit tests for critical components of your CRM system to catch bugs early in the development process.

Integrate continuous integration (CI) tools to run tests automatically..

**4.Code Reviews:**

Conduct regular code reviews with your development team to identify and fix issues collaboratively.

**5.Version Control:**

Use version control systems like Git to track changes and easily revert to a working state if a bug is discovered.

**Traceability**

**1.Issue Tracking System:**

Use an issue tracking system (e.g., Jira, Trello, or GitHub Issues) to record and prioritize debugging tasks.

**2.Requirements and Specifications:**

Maintain a detailed record of project requirements and specifications, linking them to specific features or code components.

**3.Version Control and Commits:**

Ensure that every code commit is linked to an issue or task in your issue tracking system, making it easy to trace back to the changes made.

**4.Documentation:**

Keep comprehensive documentation that covers the system's architecture, data flow, and components.

Include references to code files and functions within the documentation.

**5.Code Comments:**

Use comments in your code to explain complex logic and reference related requirements or specifications.

**6.Collaboration and Communication:**

Encourage collaboration among team members and maintain clear communication about changes, issues, and updates in the project.

**Exception handling**

Exception handling is crucial for a robust and reliable CRM project for co-living bookings. It helps your application gracefully respond to unexpected errors and prevents crashes. Here's how to handle exceptions effectively:

**1.Identify Critical Points**:

Identify critical points in your code where exceptions could occur, such as database connections, API calls, or file I/O operations.

**2.Identify Critical Points:**

Identify critical points in your code where exceptions could occur, such as database connections, API calls, or file I/O operations.

**3.Use Try-Catch Blocks**:

Wrap code sections that may throw exceptions in try-catch blocks. Catch specific exception types to handle them appropriately.

**4.Custom Exceptions**:

Define custom exception classes to represent application-specific errors. This can make your exception handling more expressive.

class CustomException(Exception):

def \_\_init\_\_(self, message):

super().\_\_init\_\_(message)

**5.Logging and Reporting**:

Log exceptions with detailed information, including stack traces, to help with debugging. You can use logging libraries like Log4j, Log4net, or the built-in logging module in Python.

**6.User-Friendly Error Messages**:

Display user-friendly error messages when exceptions occur. Avoid exposing sensitive system details to users.

**7.Alerts and Notifications:**

Set up alerts or notifications (email, SMS, or system alerts) for critical exceptions, allowing the development team to respond quickly to issues.

**8.Error Page or UI**:

Create a custom error page or user interface to inform users about errors gracefully, provide guidance, and collect feedback if necessary.

Effective exception handling enhances the reliability and usability of your CRM system, making it more resilient in the face of unexpected issues while providing a better experience for users.