This section will detail the full functional requirements for both applications. Since this is a project for a financial institution, it is important to have clearly defined functionalities that are documented in a rigid manner as typically seen in the Waterfall Methodology [potentially touch on the fact that this project will employ a SCRUM approach. However this may not be the best place to place that information]

At a high level, the requirements for both applications can be summarized as follows:

**Application 1 Client Onboarding & Data Analysis**: The application should allow prospective home buyers to log onto the platform and submit an application to purchase a home under a rent-to-own agreement. The resulting data from these applications should be aggregated and displayed on the admin user's dashboard. The admin user should be able to query the data and visualize various trends and patterns.

**Application 2:** The application should allow admin users to view and approve expression of interest forms that have been submitted by clients (prospective home owners). Once a client has been approved, the admin should be able to allocate a home to them on the system. From there, the admin should be able to monitor a clients monthly payments and adjust the rates that for their payment plan. The client should also be able to view and track their payment progress, as well as submit any complaints/issues to GCB capital via the application.

[provide more information on application 2 once client interviews are concluded]

**2.4.1 Application 1 - Client Onboarding and Data Analysis Sub-System**

The full functional requirements for the first application are detailed below with each functionality being assigned a priority. “High Priority” indicates that the functionality is essential to this iteration of the application and must be implemented. “Medium Priority” indicates that the feature although relevant, need only be implemented if time allows. “Low Priority" indicates that the functionality would be a pleasant addition, but need not be implemented for this project.

Kindly note that henceforth, the term “platform” will be used to refer to the application

|  |  |  |
| --- | --- | --- |
| **ID** | **Functional Requirement** | **Priority** |
| **FR001** | The platform authentication and registration system must allow potential home buyers to create an account with the following details: email address, password, Ghana card number, and phone number. | ● High |
| **FR002** | The authentication and registration system must require the user to verify and activate their account. A confirmation email will be sent to the email address that was used to register on the platform. | ● High |
| **FR003** | The application validation system must allow potential home buyers to submit a form/request expressing interest to purchase a home: The form should collect information on the following: personal information, physical address, employment details, bank details, and preferred location of property. [give a full breakdown in an appendix or list theme here] | ● High |
| **FR004** | The application validation system must allow potential home buyers to submit joint applications for the purchase of a home. | ● High |
| **FR005** | The authentication system must ensure that all potential home buyers applying via a joint application hold registered accounts on the system. | ● High |
| **FR006** | The registration authority should allow the creation of one “super admin” account which can create other accounts with admin privileges. This requires the super user to input a valid email address that is to be associated with the new admin account. | ● High |
| **FR007** | The verifier must allow admin users to update the inquiry application form(s) listed in FR003 and add more required fields. | ● High |
| **FR008** | The application validation system must send a confirmation email to potential home buyers once their applications have been received and logged into the database. | ● High |
| **FR009** | The platform must allow admin users to individually view applications that have been submitted by prospective home buyers. | ● High |
| **FR010** | The platform must present the following statistics and metrics to admin users: • total applications received • total applications received within a specified period • total number of organizations/employers represented in the system • average salary level. | ● High |
| **FR011** | The platform must allow admin users to view data trends by using charts and graphs to visualize the metrics in FR009. These charts and graphs must illustrate changes in the metrics over a specified period of time. | ● High |
| **FR012** | The platform must allow admin users to sort applicant data into groups, ascending, or descending order using the following fields: • name • organization/employer • preferred location • salary level | ● High |
| **FR013** | The platform must allow admin users to input data collected from real estate developers into the system. The data may include: developer's name, location of development, price point and currency, type of home, land size, and number of units. | ● High |
| **FR014** | The platform must display statistics and metrics relevant to the data that is collected from developers: these statistics include:  • number of housing units in the system • average price of houses • average number of bedrooms • number of housing developments per location. | ● High |
| **FR015** | The platform must allow admin users to sort developer data into groups, ascending, or descending order using the following fields: • developer name • number of units • number of bedrooms • price per unit. | ● High |
| **FR016** | The platform should allow the admin user to display developer sites/properties and their details on a google maps interface. | ● Medium |
| **FR017** | The AI model should allow admin users to query the database using natural language prompts and the system should return the data in a tabular and/or paragraph-based format. | ● Medium |
| **FR018** | The platform should allow admin users to import customer data from CSV files. These CSV files should be used to populate the appropriate database tables. | ● Low |

**[potentially add another table indicating functionality groups]**

**2.4.2 Application 2 – Rent-to-Own Management Sub-System**

The full set of functional requirements for the second application are detailed below. Similar to the previous section, each functionality is assigned a priority indicating whether or not it will implemented during the course of this project.

Kindly note that from here on out, prospective home owners/clients shall be referred

|  |  |  |
| --- | --- | --- |
| **ID** | **Functional Requirement** | **Priority** |
| **FR001** | The authentication system must allow potential home buyers to log in with the same account that was created and used on the client onboarding platform | ● High |
| **FR002** | The authentication system must allow admin users to login with the same account that is used to log into the admin panel of the client onboarding platform. | ● High |
| **FR003** | The platform must allow admin users to individually view applications that have been submitted by prospective home buyers. | ● High |
| **FR004** | The platform must allow admin users to reject or accept applications that have been submitted by prospective home buyers | ● High |
| **FR005** | The platform must notify clients, via email, of their acceptance or rejection in regard to their application for the rent to own scheme. The email should contain a link to the new platform. | ● High |
| **FR006** | The platform must allow admin users to allocate a home to tenants on the platform once their applications have been approved | ● High |
| **FR007** | The platform must notify tenants via email once they have been allocated a home on the platform. | ● High |
| **FR009** | The platform must display to admin users, the full details regarding payment and property ownership for each client that has been assigned a home. These details include: • initial value of the property • terminal value of the property • past and upcoming payments • amount due in the upcoming month • total amount due for the year • total value of late payments and acquired interest • total equity held by the client • total equity held by GCB Capital | ● High |
| **FR010** | The platform must display to tenants, in a simple and user friendly manner, full details regarding payment and property ownership for the home they have been assigned. These details include: • initial value of the property • terminal value of the property • past and upcoming payments • amount due in the upcoming month • total amount due for the year • total value of late payments and acquired interest • total equity held by the client • total equity held by GCB Capital | ● High |
| **FR011** | The platform must display to both admin and tenants, the complete payment schedule for the duration of the rent-to-own agreement | ● High |
| **FR012** | The platform must allow admin users to view data trends by using charts and graphs to visualize the following metrics over a specified period of time: **­­**• total monthly income for each developer • total number of assigned homes • [indicate more metrics] | ● Medium |
| **FR013** | The platform must display to admin users the following general metrics and statistics: • total income expected for the month • list of all tenants and their expected payments for their month • total income payment overdue [indicate more metrics] | ● High |
| **FR014** | The platform must allow admin users to track and view the date on which a property is handed over to a tenant | ● High |
| **FR015** | The platform must allow tenants to submit complaints regarding the property to GCB via the platform | ● High |
| **FR016** | The platform must allow tenants to make payments towards their allocated home through MoMo or Direct bank transfers. | ● Low |
| **FR017** | The platform should allow the admin user to display assigned and vacant properties and their details on a Google Maps interface. | ● Medium |

**2.5 Non-Functional Requirements**

This section details the non-functional requirements for both applications. Both systems, are being developed for a financial institution and are expected to be implemented with high standards. As such, the non-functional requirements for both applications are the same as they reflect the overarching performance and reliability criteria that are essential for applications of this nature [reword to improve readability].

The non-functional requirements are as follows:

|  |  |  |
| --- | --- | --- |
| **ID** | **Non-Functional Requirement** | **Explanation** |
| **NFR001** | Security | The application will be handling sensitive financial data. As such, it needs to adhere to industry standard practices and ensure that data is stored and transmitted as securely as possible. The application should also be resistant to common security threats such as SQL injection and Session Hijacking |
| **NFR002** | Scalability | The user base of the application is expected to grow over its first year of public use. As such, the application should be able to cope with increasing demand without any noticeable impact on the performance or reliability of the application |
| **NF003** | Maintainability | The application should be written and architected in a manner that makes it easy for developers to maintain and update the system. |
| **NF004** | Portability | The architecture and final packaging of the system should lend itself well to being deployed on various forms/types of infrastructure. E.g Should be able to transition from a cloud based deployment to privately owned infrastructure and vise versa with little to no issue |
| **NF005** | Regulatory Compliance | Since the application is to be used by a Ghanaian Financial Institution, It is important to ensure that the application complies strictly with all guidelines laid out by the Data Protection Act ,2012 (Act 843) |
| **NF006** | Robustness | The application should adequately and elegantly handle user input errors and prompt the user to enter the correct information |
| **NF007** | Responsiveness | The application should provide a smooth and problem free user experience. It should react quickly to user interaction and adjust its interface appropriately to accommodate common desktop screen sizes |
| **NF008** | Interoperability | Although the applications are being treated as separate, they should be architected and implemented in a manner that will allow them to be integrated together at a later date. This requires both applications to have well-documented RESTful APIs. |