

Table of Contents

Application Information & Rules	1
Background.....	1
Purpose	1
Project Links	1
Features & Menu Overview	2
1. Freelancer Management (Main Features)	2
2. Project Management (Complementary Features).....	2
3. Budget Management (Complementary Features).....	3
Data Structures	3
1. Freelancer Data Structure.....	3
2. Project Data Structure	4
3. Company Budget Data Structure	5
Important Application Rules & Assumptions	6
1. Budgeting System	6
2. Freelancer Availability.....	6
3. Project Cost Calculation	6
4. Data Management	7
Terminology	7
Flowcharts with Explanations.....	7
Application Main Menu	7
Freelancer Management Menus	9
1. Main Menu	9
2. Hire New Freelancer	10
3. Review Freelancer Profiles	11
4. Search Freelancer	12
5. Update Freelancer Information.....	13
6. Fire Freelancer	14
7. View Freelancers Performance Report	15
Project Management Menus.....	16
1. Main Menu	16
2. Review Projects	17
3. Assign Project to Freelancer	18
4. Mark Project as Completed	19
5. Cancel Project.....	20
Budget Management Menus	21
1. Main Menu	21
2. Adjust Budget	22

Application Information & Rules

Author: Rio Pramana – DTIDSOL-02

Module 1 Capstone Project – Freelancer Management System (FMS)

Background

The **Freelancer Management System (FMS)** is a **CRUD-based terminal application** built using **Python fundamentals**. This application simulates a **real-world freelancer management system**, allowing a company to **hire, manage, and track freelancers and projects** efficiently.

Purpose

The **FCMS** is designed to help businesses:

1. **Manage freelancers efficiently** – Track freelancer information, assign projects, and review performance.
2. **Handle project assignments & budgets** – Ensure that company spending aligns with available resources.
3. **Automate financial tracking** – Monitor freelancer costs and project expenses dynamically.

Unlike simple CRUD applications, **FCMS mimics real business operations**, integrating **budget control and filtering functionalities** to enhance decision-making.

Project Links

1. Project Google Drive (all files in one folder): [Click This Link](#)
2. Draw.io files: [Click This Link](#)
3. Flowchart images: [Click This Link](#)
4. Entity Relationship Diagram: [Click This Link](#)

Features & Menu Overview

The FMS consists of **three core management sections**:

1. Freelancer Management (Main Features)

a. **Hire Freelancer** – (*Create*)

Adds a new freelancer with basic info such as name, skills, and hourly rate.

b. **Review Freelancer Profiles** – (*Read, with filtering options*)

Displays all freelancers, filterable by freelancer status (available/assigned).

c. **Search Freelancer** – (*Search function*)

Allows searching freelancers by **name** (partial match), **skills** (comma-separated, partial match, OR logic), or **ID** (exact match).

d. **Update Freelancer Information** – (*Update*)

Edits freelancer details with validation.

e. **Fire Freelancer** – (*Delete*)

Removes selected freelancers who **have no active projects**.

f. **View Freelancers Performance Report** – (*Read, with sorting options*)

Displays **total completed projects & earnings** with option to **sort freelancers** based on earnings.

2. Project Management (Complementary Features)

a. **Assign Freelancer to Project** – (*Create + Update*)

Creates a new project while assigning a freelancer to work on it (with budget validation mechanism).

b. **Mark Project as Completed** – (*Update*)

Finalizes a project, updates freelancer availability, company's budget and allocated funds.

c. **Review Projects** – (*Read, with filtering options*)

Displays all projects, filterable by project status (active/completed).

d. **Cancel Project** – (*Delete*)

Releases allocated funds if a project is cancelled before completion.

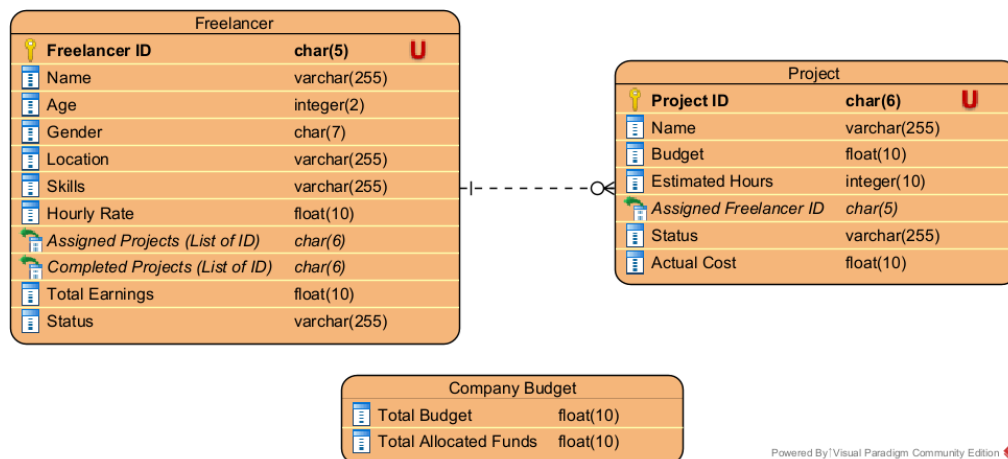
3. Budget Management (Complementary Features)

a. **Adjust Budget** – (*Update*)

Allows the user to **increase or decrease** the company's budget, ensuring it **never falls below allocated funds**.

Data Structures

This section outlines the **data structures** used in the Freelancer Management System (FMS). Since this project is built strictly using **Python collections**, we model entities as **nested dictionaries and lists** instead of using relational database concepts. For deeper understanding of how each entity relates to each other, please refer to [Important Application Rules & Assumptions](#) section of this document.



1. Freelancer Data Structure

Freelancers are stored in a **dictionary**, where each freelancer has a **unique ID** as the key, and their details are stored as values in a nested dictionary.

Example data:

```
freelancers = {
    "FR001": {
        "name": "John Doe",
        "age": 28,
        "gender": "Male",
        "location": "New York",
        "skills": ["Python", "Machine Learning", "Data Analysis"],
        "hourly_rate": 10.0,
        "status": "Assigned",
        "assigned_projects": ["P0001"],
        "completed_projects": ["P0002", "P0003"],
        "total_earnings": 3500.0
    }
}
```

Fields explanation:

Field	Type	Description
freelancer_id	str	Unique ID for each freelancer.
name	str	Freelancer's full name.
age	int	Freelancer's age (must be ≥ 18).
gender	str	"Male" or "Female".
location	str	Freelancer's city or country.
skills	list (str)	A list of the freelancer's skills.
hourly_rate	float	The hourly rate charged by the freelancer.
Status	str	"Available" or "Assigned".
assigned_projects	list (str)	List of currently active project IDs.
completed_projects	list (str)	List of completed project IDs.
total_earnings	float	The total earnings from completed projects.

2. Project Data Structure

Projects are stored in a **dictionary**, where each project has a **unique ID** as the key, and project details are stored as values in a nested dictionary.

Example data:

```
projects = {  
    "P0001": {  
        "name": "AI Chatbot Development",  
        "budget": 2000.0,  
        "estimated_hours": 40,  
        "assigned_freelancer": "FR001",  
        "status": "Active",  
        "actual_cost": 400.0  
    }  
}
```

Fields explanation:

Field	Type	Description
project_id	str	Unique ID for each project.
name	str	Project name.
budget	float	Total budget allocated for the project.
estimated_hours	int	Estimated hours required to complete the project.
assigned_freelancer	str	Freelancer ID assigned to the project.
status	str	"Active", "Completed", or "Cancelled".
actual_cost	float	The actual project cost, set after completion.

3. Company Budget Data Structure

The **company's budget** is stored in a dictionary containing **two key fields**: `total_budget` and `total_allocated_funds`.

Example data:

```
company_budget = {  
    "total_budget": 10000.0,  
    "total_allocated_funds": 4000.0  
}
```

Fields explanation:

Field	Type	Description
project_id	str	Unique ID for each project.
total_budget	float	The company's total available funds.
total_allocated_funds	float	Funds currently allocated to active projects.

Important Application Rules & Assumptions

1. Budgeting System

- The app's budgeting system consists of **two parts: Company budget & Allocated funds**.
- Hiring a freelancer does not affect any budget/funds**, it only adds the freelancer data to the company's database for management. Only project completion/cancellation affect budget & funds.
- Company budget** is pre-determined and changeable. It is **reduced only when a project is completed** (not at assignment).
- Allocated funds represent money reserved for ongoing/active projects**. It is **reduced only when a project is completed or cancelled**.
- The budget cannot be decreased below allocated funds. Allocated funds cannot be greater than the budget.

2. Freelancer Availability

- Freelancers can **only take one project at a time**. One project can only be assigned to one freelancer.
- Once a project is completed or cancelled, the freelancer **becomes available again**.

3. Project Cost Calculation

- The user **sets a maximum budget & estimated completion hours** when creating a project.
- Project actual cost = Freelancer hourly rate × Estimated hours**.
- The user **can only select freelancers with actual cost less than the maximum budget**.

- d. If the company can't afford any freelancer, the admin must **increase the project's maximum budget or cancel the project creation.**
- e. **Every project is fully paid only after completion.**

4. Data Management

- a. The system uses **Python dictionaries** as the primary data storage format.
- b. Sorting & filtering **are integrated into the relevant menus.**

Terminology

Term	Definition
Freelancer	A worker who can be hired to complete projects.
Project	A task assigned to a freelancer with a budget and duration (in hours).
Company Budget	The total money available for freelancer payments.
Allocated Funds	Money reserved for ongoing projects (not yet deducted).
Project Cost	The total amount a freelancer will be paid upon project completion.
Active Project	A project currently in progress (not completed or cancelled).

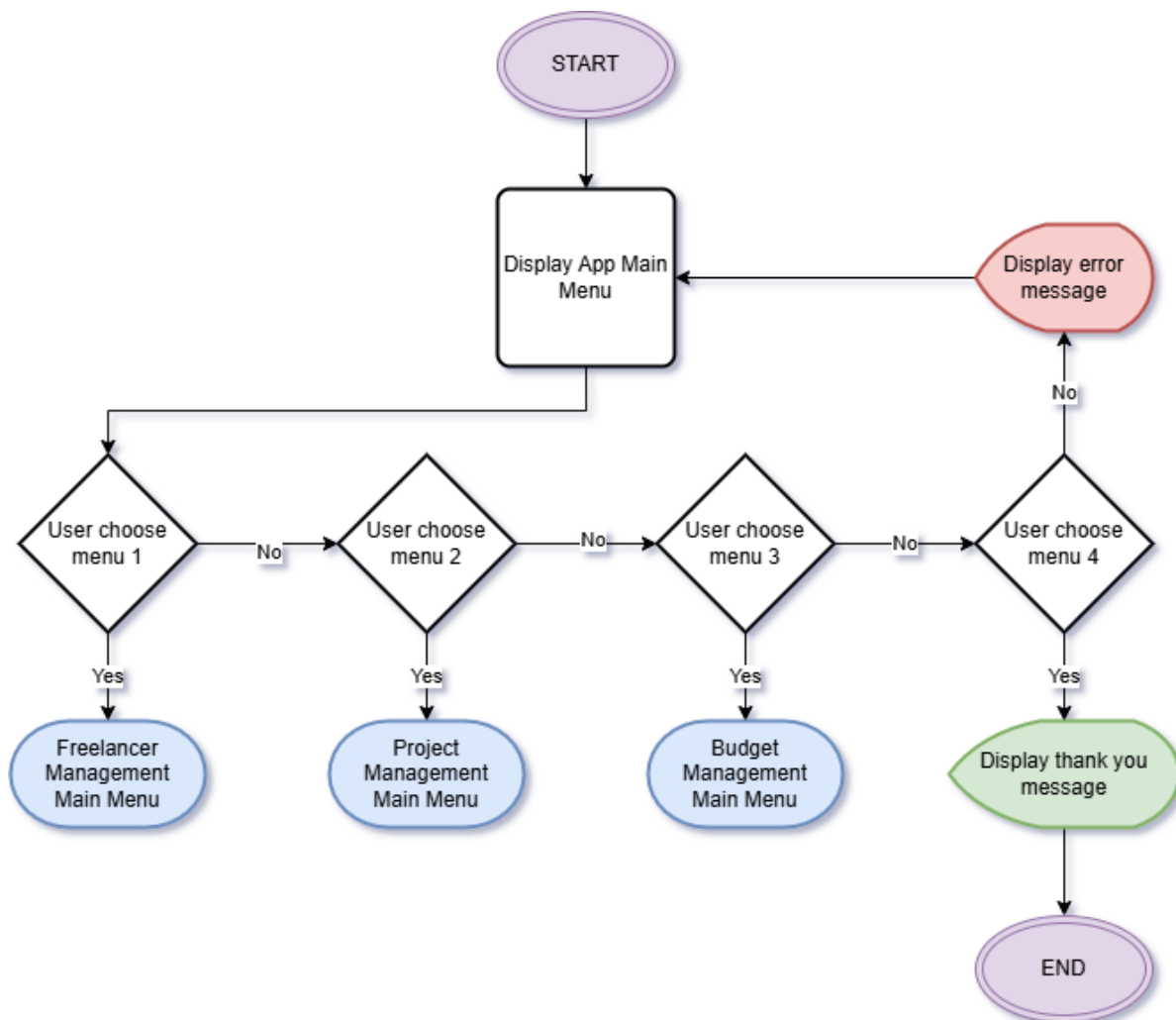
Flowcharts with Explanations

Application Main Menu

This is the **starting point** of the application. The admin is presented with the core management options:

1. **Freelancer Management**
2. **Project Management**
3. **Budget Management**
4. **Exit Application**

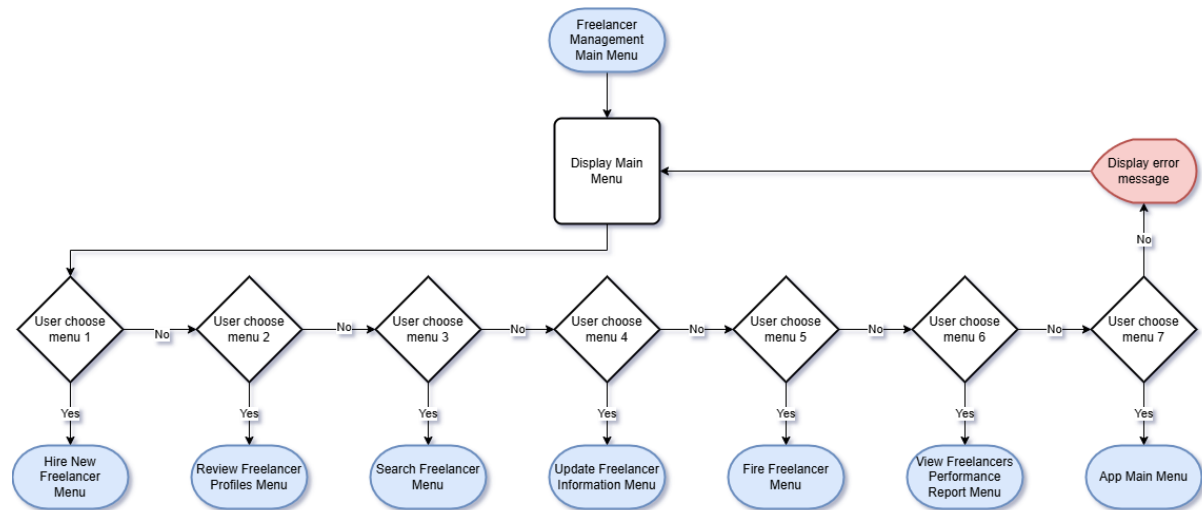
The admin selects an option, which directs them to the corresponding submenu. If an invalid option is chosen, an error message is displayed, prompting a retry. Image link: [Click Me](#)



Freelancer Management Menus

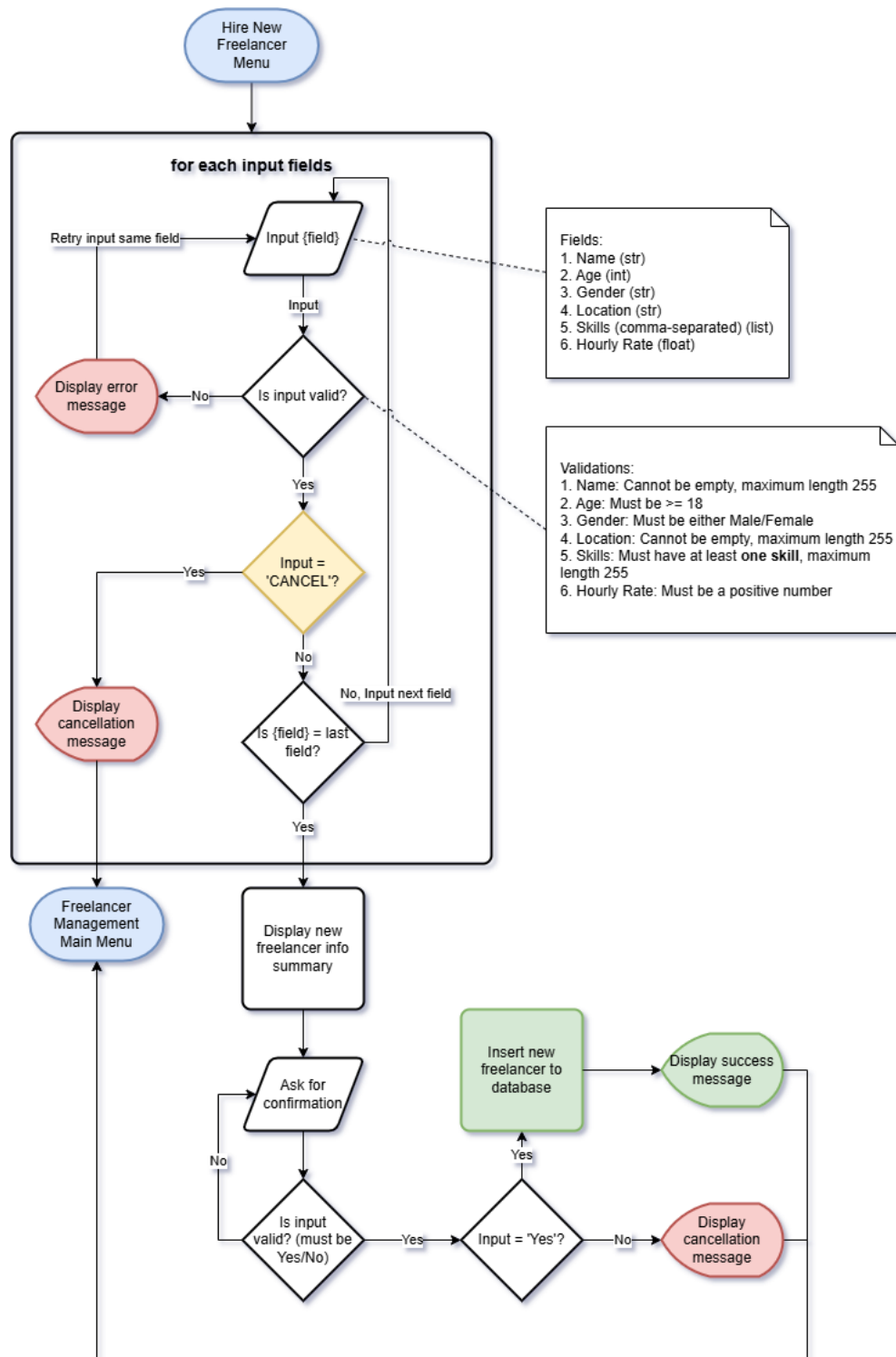
1. Main Menu

This menu serves as the hub for managing freelancers. The user can do all CRUD functions related to freelancers here. Image link: [Click Me](#)



2. Hire New Freelancer

The admin inputs a freelancer's **basic details**. Each field is validated before proceeding. If the input is invalid, the admin is prompted to correct it. After all fields are entered, a **confirmation step** allows reviewing before saving the freelancer. Image link: [Click Me](#)

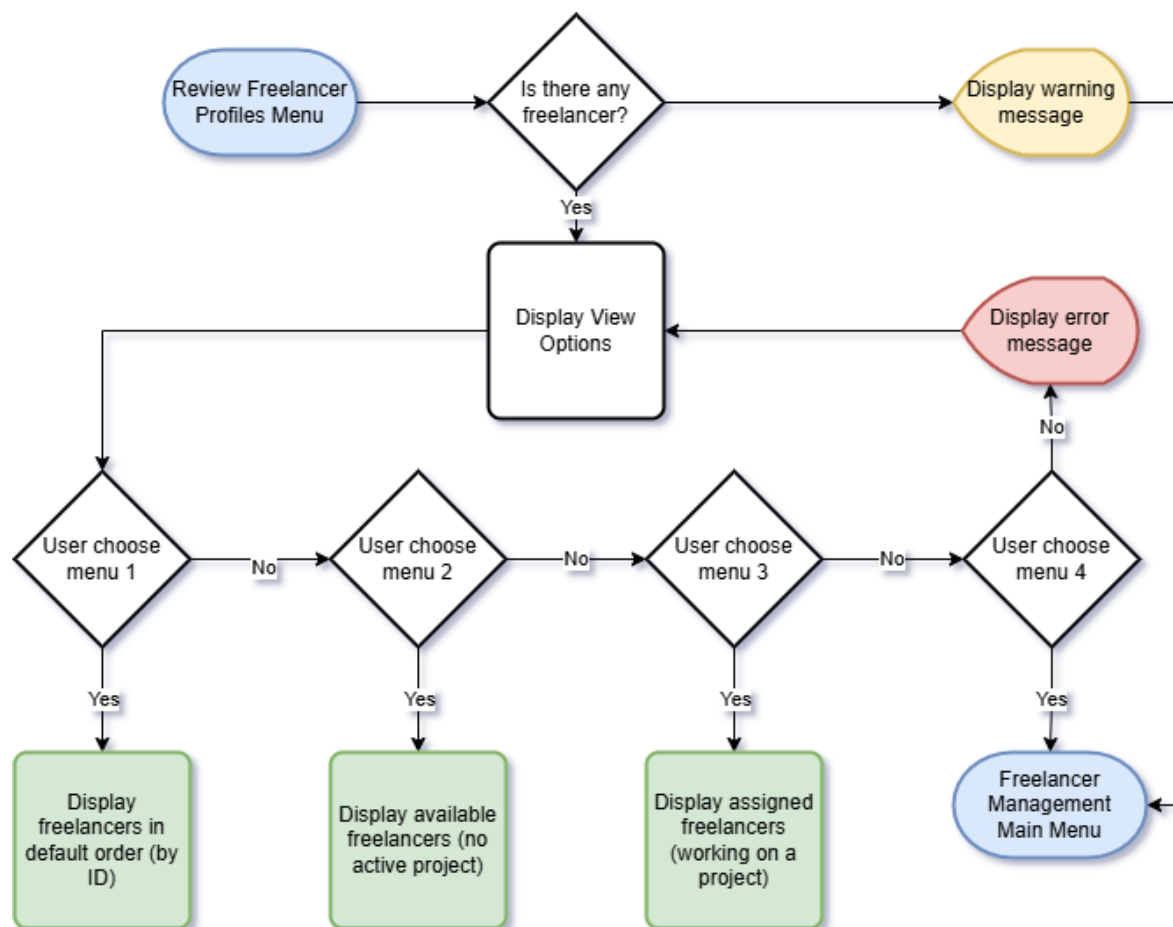


3. Review Freelancer Profiles

This menu allows the admin to view freelancers based on different categories:

- **All freelancers** (sorted by ID)
- **Available freelancers** (those without an active project)
- **Assigned freelancers** (currently working on a project)

Only essential freelancer details (e.g. ID, Name, Skills, Hourly Rate, Availability) are displayed in this section. Image link: [Click Me](#)

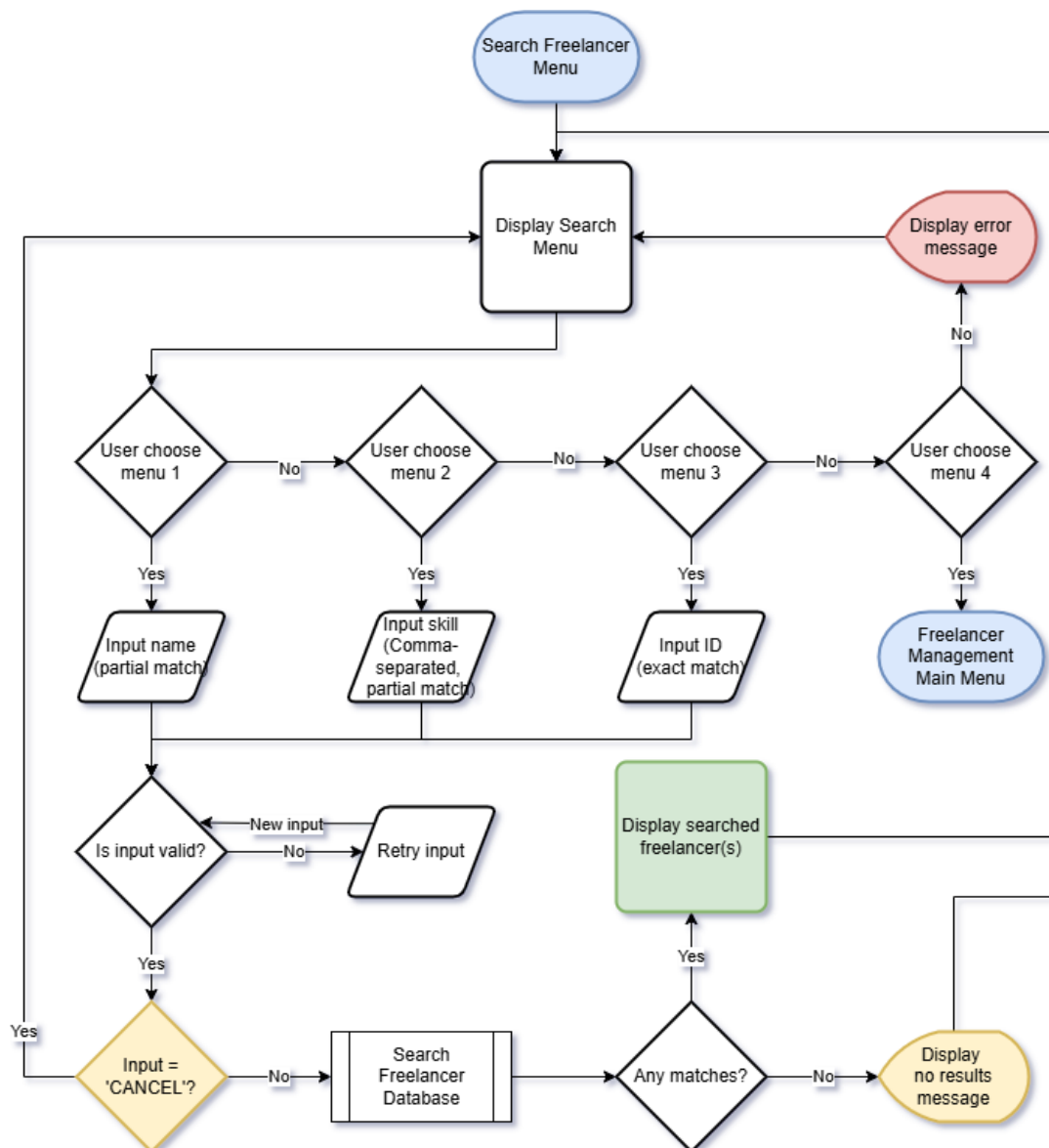


4. Search Freelancer

The admin can search freelancers by:

- **Name (partial match)**
- **Skills (comma-separated, OR logic for multiple skills)**
- **Freelancer ID (exact match)**

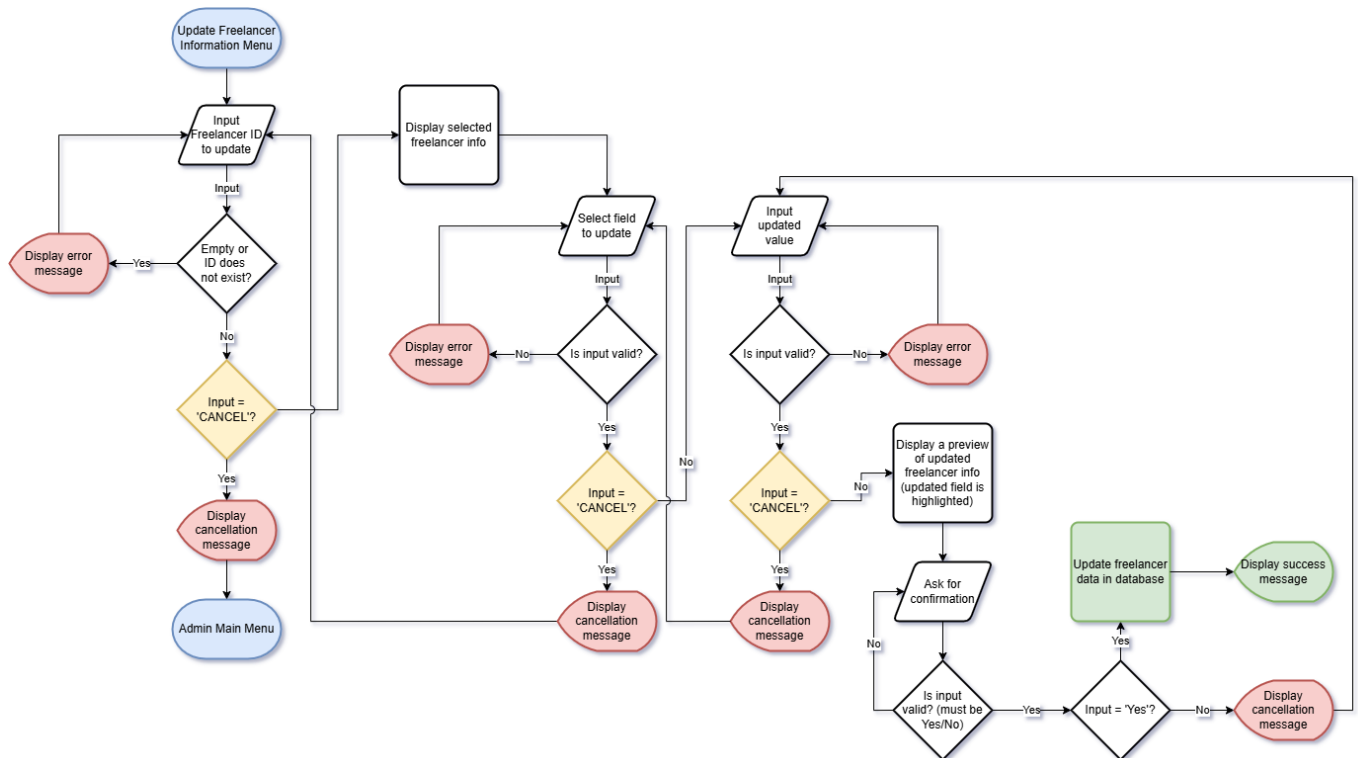
If no freelancer is found, an error message is displayed, and the admin can retry or exit the search. Image link: [Click Me](#)



5. Update Freelancer Information

The admin selects a freelancer by entering their **ID**, after which their **current details** are displayed. The admin then chooses which field to update. The new input is validated before applying changes. A final **confirmation step** ensures correctness before updating the database.

Image link: [Click Me](#)

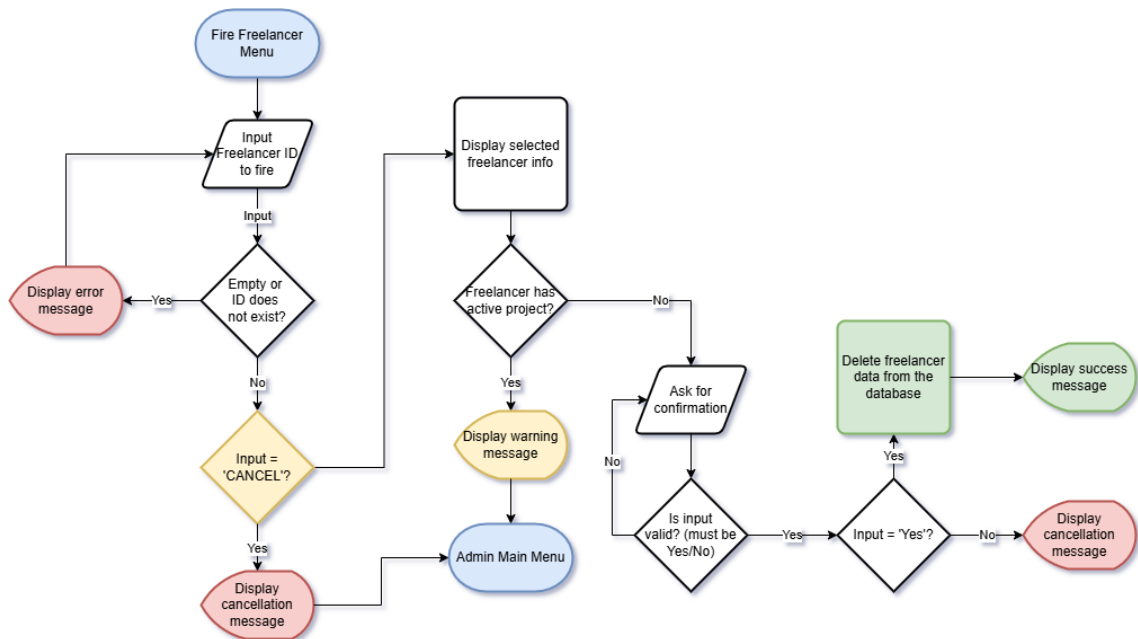


6. Fire Freelancer

The admin enters the **Freelancer ID** to remove them.

- If the freelancer has an **active project**, firing is **not allowed**.
- If the freelancer has no active project, a **confirmation step** is required before deletion.

Image link: [Click Me](#)

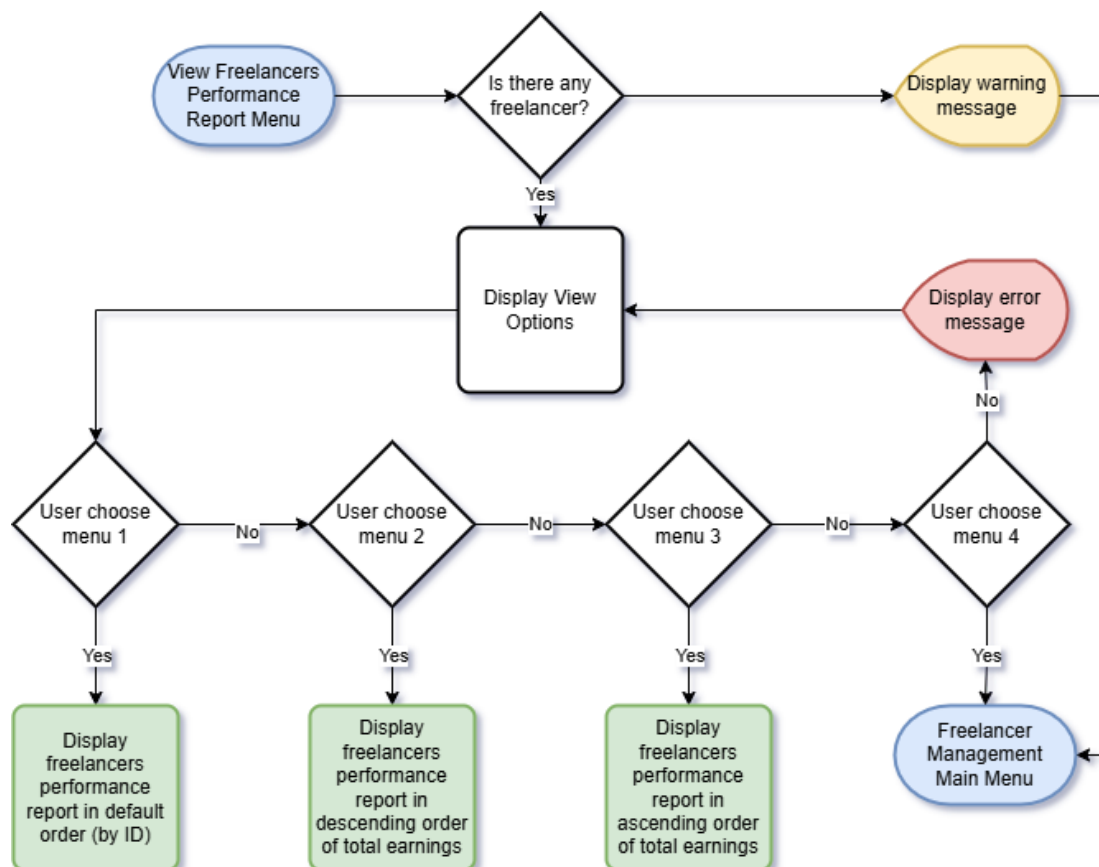


7. View Freelancers Performance Report

This report provides insights into freelancer earnings and project completion history. The admin can:

- View freelancers report sorted by ID (default).
- View freelancers report sorted by **total earnings (ascending or descending)**.

Image link: [Click Me](#)



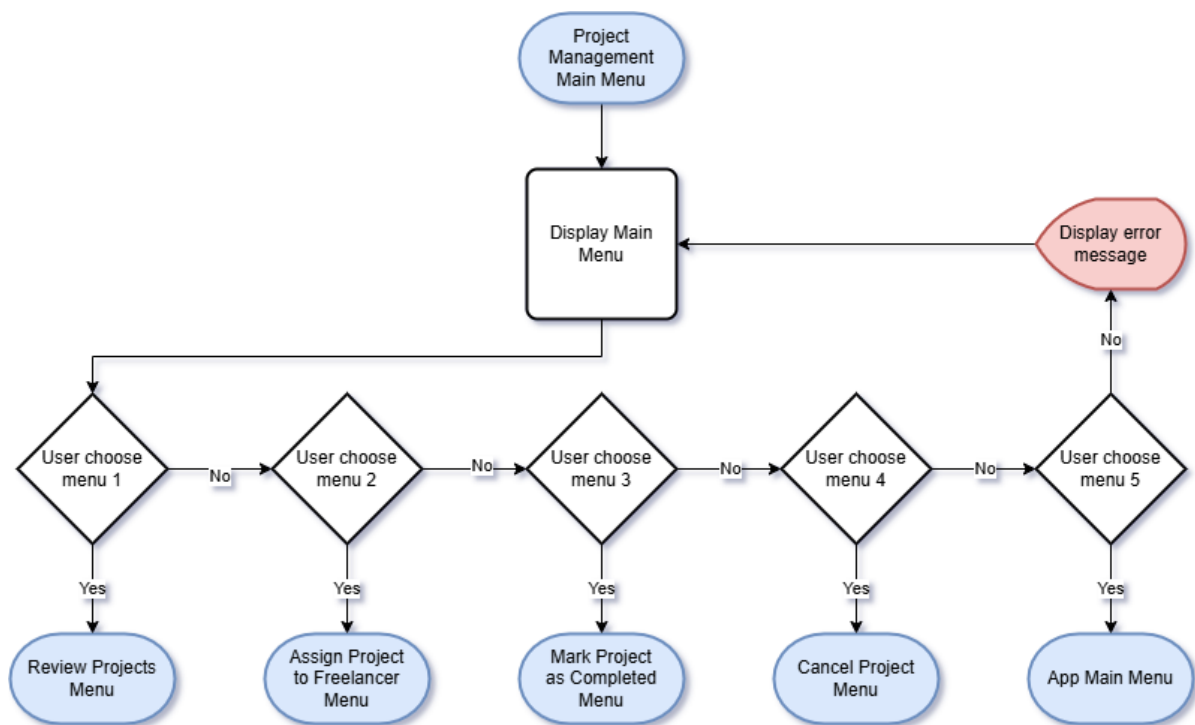
Project Management Menus

1. Main Menu

This menu provides access to all project simple CRUD actions that are related to the app's purpose for freelancer management, including:

- Assigning a project to a freelancer
- Marking a project as completed
- Reviewing projects
- Cancelling projects
- Returning to the **Application Main Menu**

Image link: [Click Me](#)

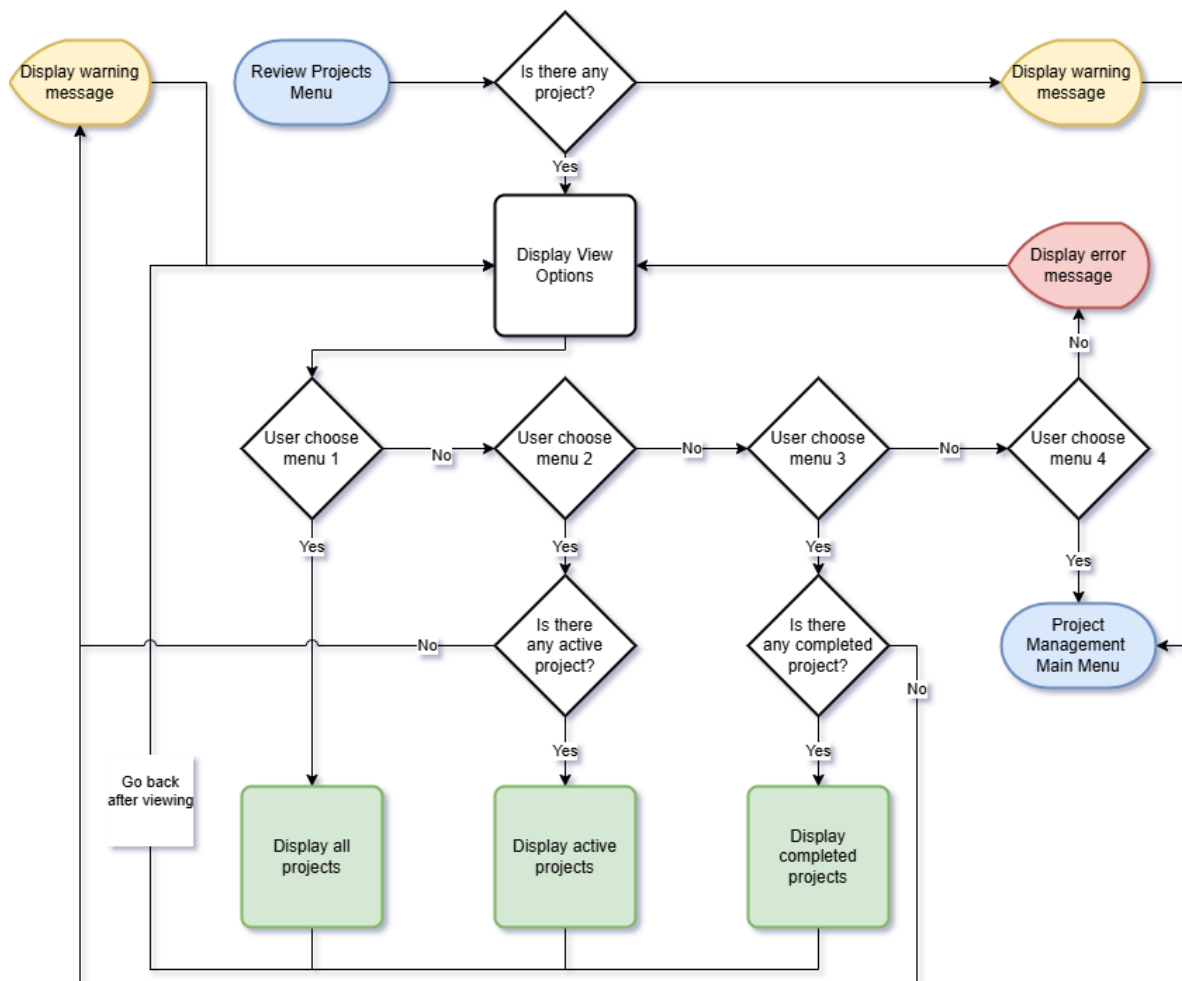


2. Review Projects

The admin can view and filter projects by:

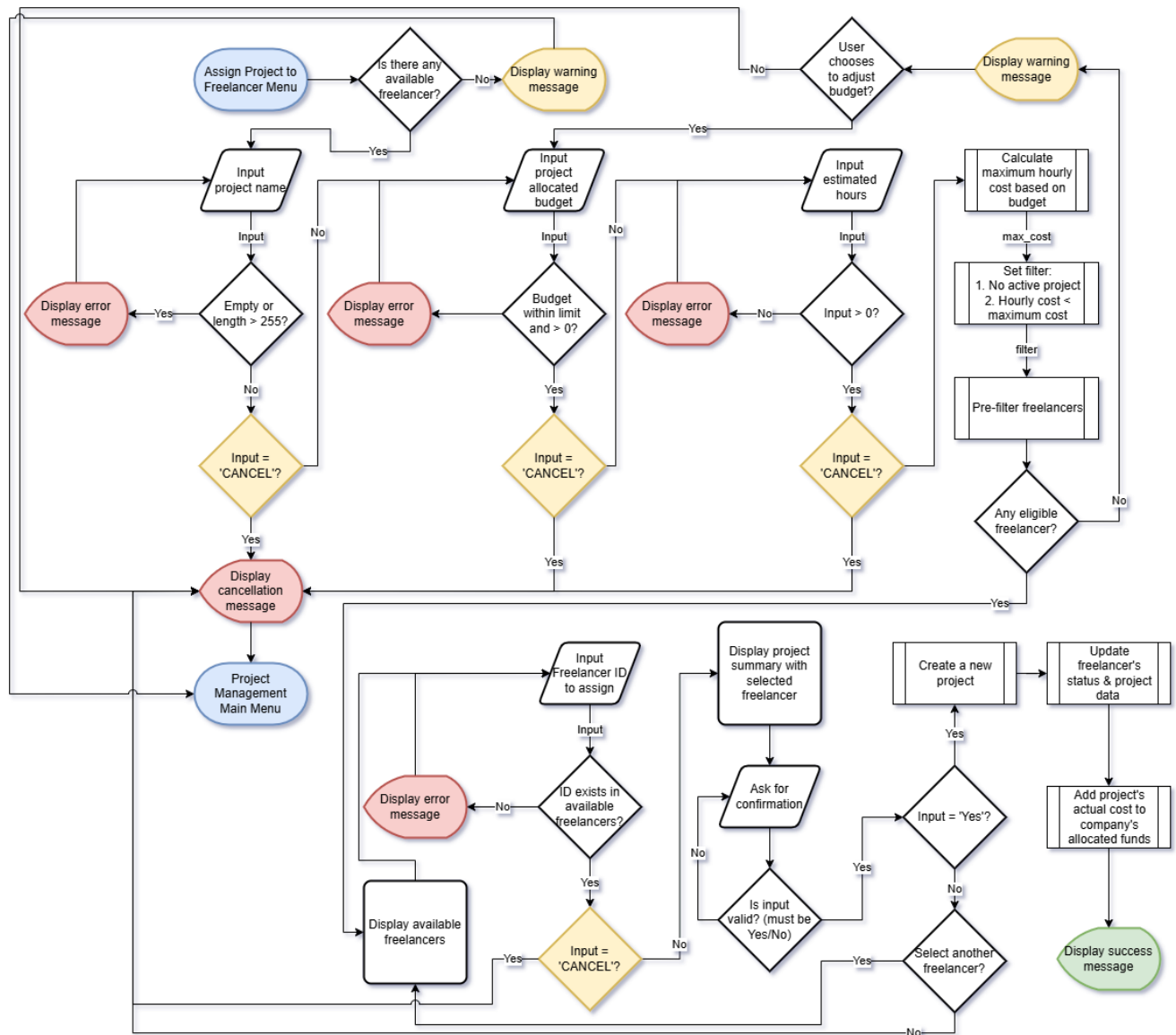
- **All Projects** (no filter)
- **Active Projects**
- **Completed Projects**

Image link: [Click Me](#)



3. Assign Project to Freelancer

The admin creates a project, entering its details, including name, maximum budget, and estimated hours. A **pre-filtering system** ensures that only **freelancers who are available and whose hourly rate fits within the budget** can be assigned. If no eligible freelancer is found, the admin must **adjust the budget or cancel**. Image link: [Click Me](#)

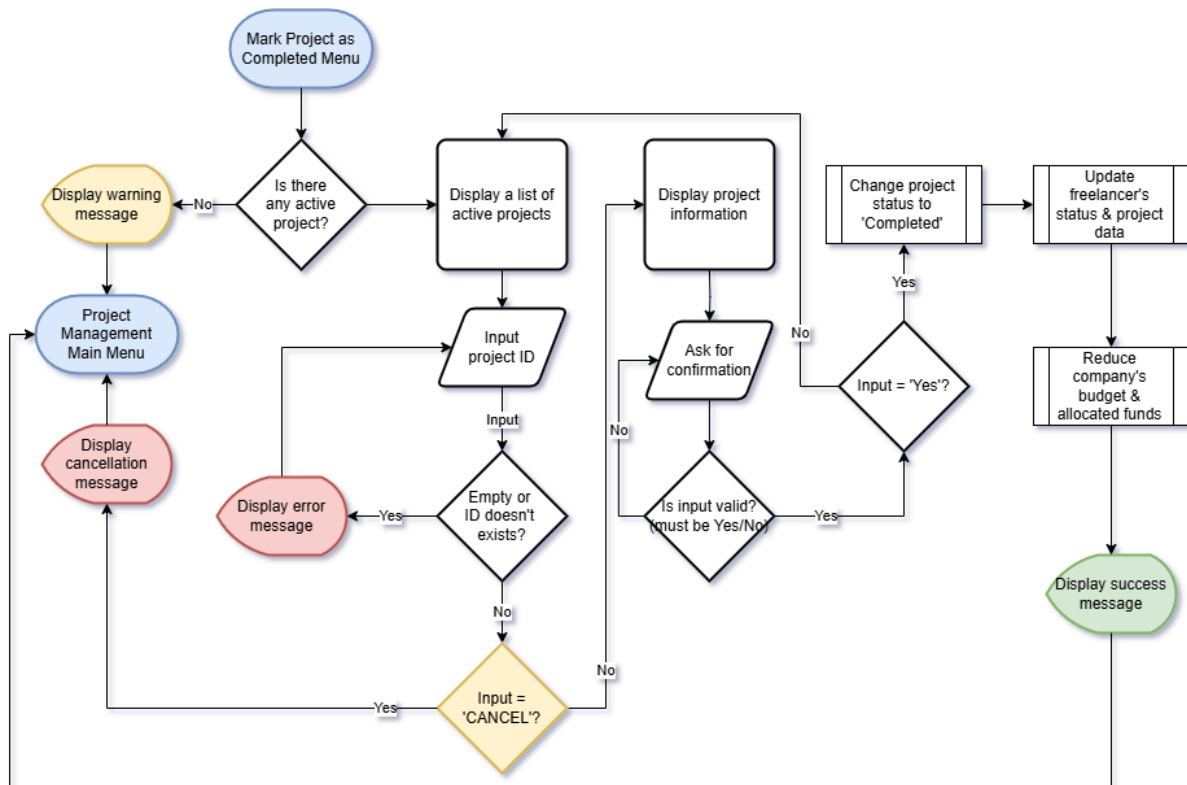


4. Mark Project as Completed

The admin selects a **project to mark as completed**.

- The **freelancer becomes available** for new assignments.
- The **company budget & allocated funds are reduced** by the project's actual cost.

Image link: [Click Me](#)

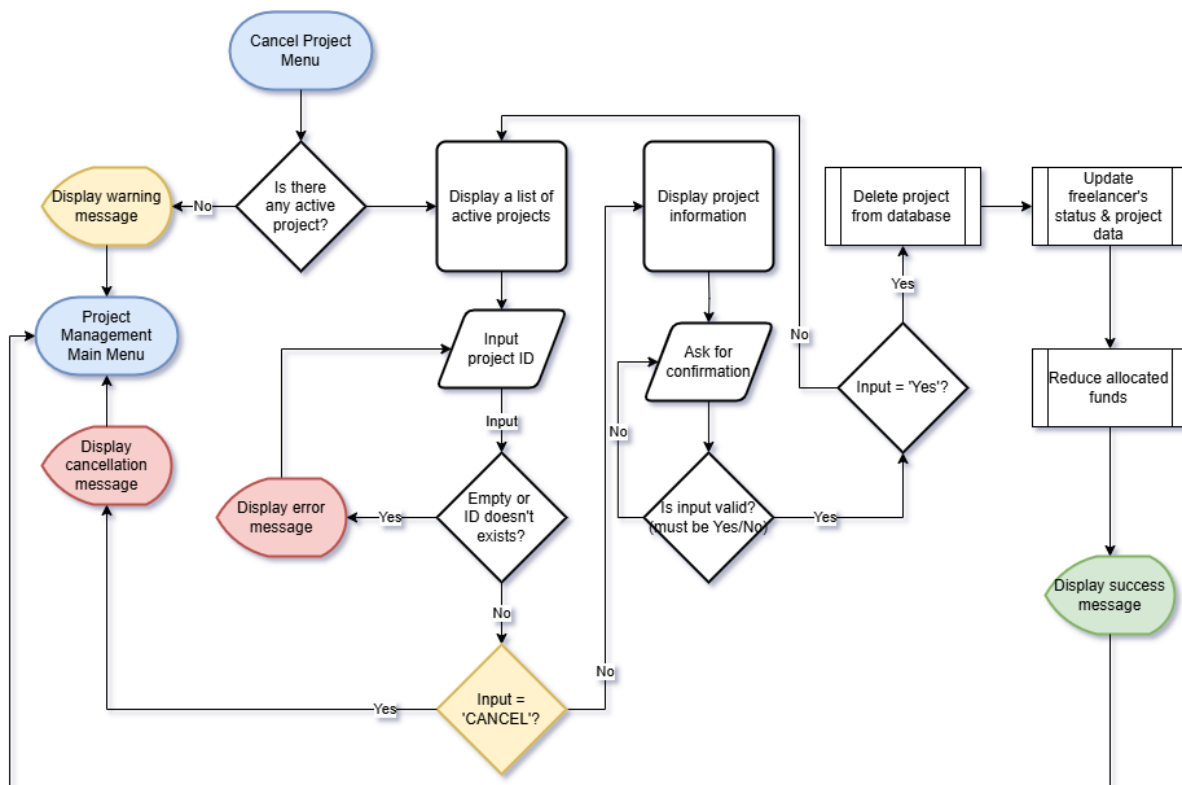


5. Cancel Project

The admin can cancel a project.

- **Allocated funds are freed up** if the project is cancelled.
- **Cancelled projects will be deleted from the database.**
- **The freelancer assigned to the project becomes available again.**

Image link: [Click Me](#)



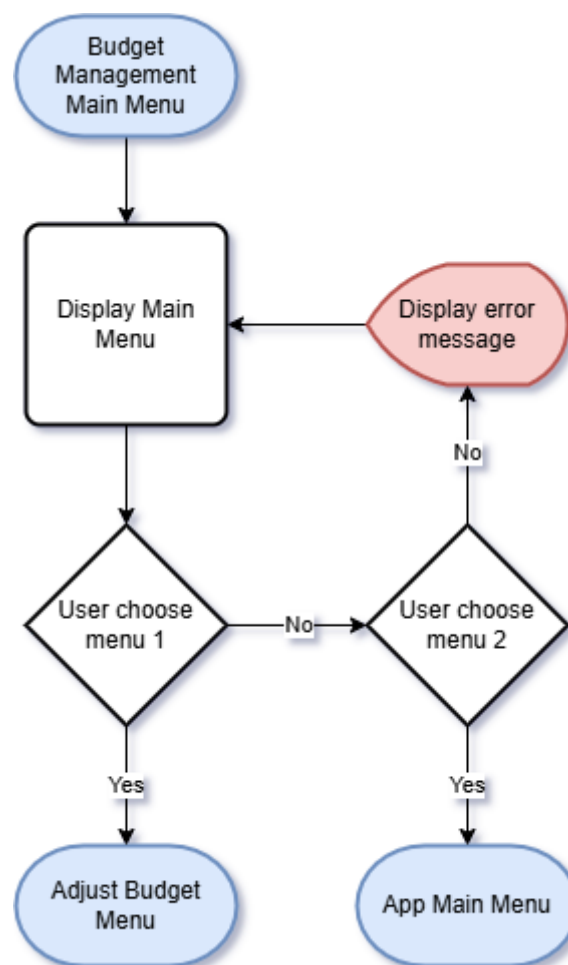
Budget Management Menus

1. Main Menu

This section allows the admin to **adjust the company budget**. The admin can:

- View the current budget and allocated funds
- Increase or decrease the budget
- Return to the **Application Main Menu**

Image link: [Click Me](#)



2. Adjust Budget

This menu allows the admin to **increase or decrease** the company's budget.

- The budget **cannot be set below total allocated funds** (the amount already committed to active projects).
- The new budget cannot be the same as the old budget.
- A **confirmation step** ensures changes are intentional.

Image link: [Click Me](#)

