# Chapter 1: Problem Specification

## 1. Requirements Gathering Process

To gather requirements for the Budget Management App, I used a mixed-method approach, combining personal observation with informal interviews conducted with 6 individuals from varied backgrounds, including university students, part-time employees, and young professionals.  
  
The process was structured as follows:  
  
- Step 1: Observation  
 I recorded common budgeting issues over a 2-week period among peers and myself. I noted difficulties in tracking spending, forgotten transactions, and manual effort involved in using spreadsheets.  
  
- Step 2: Informal Interviews  
 I prepared a short set of open-ended questions and asked each participant about their budgeting habits, frustrations, and desired app features. Sample questions included:  
 • How do you currently manage your monthly expenses?  
 • What problems do you face when tracking your budget?  
 • What would an ideal budget app look like for you?  
  
- Step 3: Pattern Analysis  
 Responses were grouped and analyzed manually. Patterns identified:  
 • Users often forget to log expenses in real-time.  
 • There’s a high demand for a mobile-friendly and minimal interface.  
 • Users want to set a monthly cap and get notified when approaching it.  
 • Users prefer exporting summaries (PDF/CSV).  
 • Many don’t want to rely on internet access for basic use.  
  
These insights were directly translated into system requirements.

## 2. User Requirements

|  |  |  |
| --- | --- | --- |
| Code | Description | Source Reference |
| UR1 | The user should be able to add expenses with amount, category, and notes. | Interview Q2, Q3 |
| UR2 | The user should be able to set a monthly budget limit. | Interview Q3 |
| UR3 | The app should alert the user when nearing or exceeding the budget. | Observation, Q3 |
| UR4 | The user should be able to view a monthly summary of expenses. | Q1, Q2 |
| UR5 | The user should be able to edit or delete past expenses. | Interview Q3 |
| UR6 | The app should support categorizing expenses (Food, Bills, etc). | Q2, Q3 |
| UR7 | The user should be able to export expense reports to PDF or CSV. | Interview Q3 |
| UR8 | The app should be usable on mobile and desktop. | Q1, Q3 |
| UR9 | The app should work offline for core functions. | Interview Q2 |
| UR10 | The app should protect user data and not expose it publicly. | Observation |

## 3. System Requirements

### 3.1 Functional Requirements

|  |  |
| --- | --- |
| Code | Functional Requirement |
| FR1 | The system shall allow users to add new expense entries (amount, category, note, date). |
| FR2 | The system shall allow users to edit or delete previously entered expenses. |
| FR3 | The system shall allow users to set a monthly budget for a selected month. |
| FR4 | The system shall notify users if spending nears or exceeds the set budget. |
| FR5 | The system shall provide a monthly summary of categorized expenses and totals. |
| FR6 | The system shall support exporting data to PDF or CSV format. |

### 3.2 Non-Functional Requirements

|  |  |
| --- | --- |
| Code | Non-Functional Requirement |
| NFR1 | The system shall respond to user actions within 2 seconds. |
| NFR2 | The system shall be usable and readable on both desktop and mobile browsers. |
| NFR3 | The system shall store data securely using validated input and SQL protection techniques. |
| NFR4 | The system shall work offline for expense entry and budget viewing. |
| NFR5 | The user interface shall be clean, minimal, and easy to navigate. |