

**Trimester March/April, 2025**

**CSE6224 SOFTWARE REQUIREMENTS ENGINEERING**

**Project Part 1**

**Topic: Campus Ride-Sharing Platform with**

**Parking System Integration**

**Kano Model Documentation**

|  |  |  |
| --- | --- | --- |
| Name | Student ID | Course |
| Chee Rui | 1211112287 | Bachelor of Computer Science |
| Teh Li Wei | 1211109581 | Bachelor of Computer Science |
| Sow Chien Yee | 1211210800 | Bachelor of Computer Science |
| Lai Zi Xuan | 1211109451 | Bachelor of Computer Science |

Table of Contents

[1 Elicitation Strategy 3](#_Toc198085474)

[1.1 Justification for Using the Kano Model Elicitation Strategy Overview 3](#_Toc198085475)

[1.2 Classification of Requirements Using Kano Model 3](#_Toc198085476)

[2 Elicitation Execution and Findings 3](#_Toc198085477)

[2.1 Categorized Requirements (Based on Kano) 3](#_Toc198085478)

[3 Appendices 4](#_Toc198085479)

[3.1 References 4](#_Toc198085480)

1 Elicitation Strategy

1.1 Justification for Using the Kano Model Elicitation Strategy Overview

For our Campus Ride-Sharing Platform with Parking System Integration project, we selected the Kano Model as our primary requirements classification framework for several compelling reasons:

1. **Feature Prioritization Based on User Satisfaction**: The Kano Model enables us to categorize features based on their impact on user satisfaction, which is critical for a user-facing system like our ride-sharing and parking platform. This helps us focus development efforts on the most impactful features first.
2. **Clear Distinction Between Necessity and Innovation**: Our project combines essential functionality (secure login, parking availability) with innovative features (ride matching algorithms, reporting systems). The Kano Model's three-tier classification (Must-be, Satisfiers, Delighters) provides a clear framework to distinguish between these different types of requirements.
3. **Stakeholder Diversity Management**: Our system serves multiple user groups with potentially competing interests drivers want control over ride approval, while riders prefer automated matching. The Kano framework helps objectively categorize these diverse needs based on satisfaction impact rather than stakeholder influence.
4. **Effective Resource Allocation**: With limited development resources, we needed a systematic approach to determine which features to implement first. The Kano Model provides a data-driven method to allocate resources where they will have the greatest positive impact on user experience.
5. **Prevention of Feature Creep**: In complex systems like ours that combine two major functions (ride-sharing and parking management), feature creep is a significant risk. The Kano Model helps distinguish between essential features and "nice-to-haves," keeping the project scope manageable.
6. **Future Enhancement Planning**: By identifying potential "delighter" features early, we can develop a roadmap for future enhancements after the core system is launched. This approach allows for iterative development based on clear prioritization.
7. **Objective Decision-Making**: The quantitative nature of the Kano analysis (using positive/negative question pairs and evaluation grids) provides objective data for requirement decisions, reducing subjective bias in feature selection.
8. **Integration with Multiple Elicitation Techniques**: The Kano Model effectively complements our selected elicitation techniques (questionnaires, interviews, and prototyping) by providing a consistent framework to categorize findings from all these sources.

The Kano Model's structured approach to requirement classification proved invaluable in helping us navigate the complex needs of our campus ride-sharing and parking system while ensuring we deliver maximum user satisfaction with our initial implementation.

1.2 Classification of Requirements Using Kano Model

The Kano Model classifies requirements into three primary categories based on how they fulfill customer satisfaction:

**Dissatisfiers (Must-Be Requirements)**

These are basic expectations that users assume will be present in the system. Their presence does not significantly increase satisfaction, but their absence causes substantial dissatisfaction.

**Key characteristics:** Considered essential; users often don't explicitly mention them because they're assumed

**Impact on satisfaction:** Absence → Strong dissatisfaction; Presence → Neutral (no positive satisfaction)

**Examples in our system:** Login functionality, security features, basic navigation controls

**When asked about these features:**

* If present: Users typically respond with "I expect it" or "This is how it should be"
* If absent: Users respond with strong negative reactions like "I dislike it" or "This is unacceptable"

**Satisfiers (Performance Requirements)**

These features provide linear satisfaction—the better they are implemented, the greater the user satisfaction. Their absence causes some dissatisfaction, and their presence increases satisfaction proportionally.

**Key characteristics**: Considered essential; users often don't explicitly mention them because they're assumed

**Impact on satisfaction**: Absence → Strong dissatisfaction; Presence → Neutral (no positive satisfaction)

**Examples in our system**: Login functionality, security features, basic navigation controls

When asked about these features:

* If present: Users typically respond with "I expect it" or "This is how it should be"
* If absent: Users respond with strong negative reactions like "I dislike it" or "This is unacceptable"

**Delighters (Excitement Requirements)**

These are unexpected features that create delight when present but cause no dissatisfaction when absent since users don't anticipate them.

**Key characteristics:** Innovative, surprising features; users don't explicitly request them

**Impact on satisfaction:** Presence → High satisfaction; Absence → No effect (neutral)

**Examples in our system:** Advanced matching algorithms, gamification elements, special integration features

**When asked about these features:**

* If present: Users respond very positively with "I like it" or show excitement
* If absent: Users respond neutrally with "I am neutral" or "I can tolerate it"

**Classification Process**

Our classification process consisted of the following steps:

1. **Question Pairing**: For each potential feature, we created a pair of questions—one positive (feature presence) and one negative (feature absence).
2. **Response Collection**: We gathered responses using a 5-point Kano scale:
   * I like it
   * I expect it
   * I am neutral
   * I can tolerate it
   * I dislike it
3. **Evaluation Grid**: The paired responses were plotted on a Kano evaluation grid to determine the appropriate classification.
4. **Cross-Validation**: Feature classifications from questionnaires were cross-validated through interviews and prototype testing to ensure accuracy.
5. **Weighted Analysis**: In cases of mixed responses, we applied statistical analysis to determine the predominant classification based on response frequency.

**How Requirements Change Over Time**

It's important to note that requirements can shift between categories as user expectations evolve:

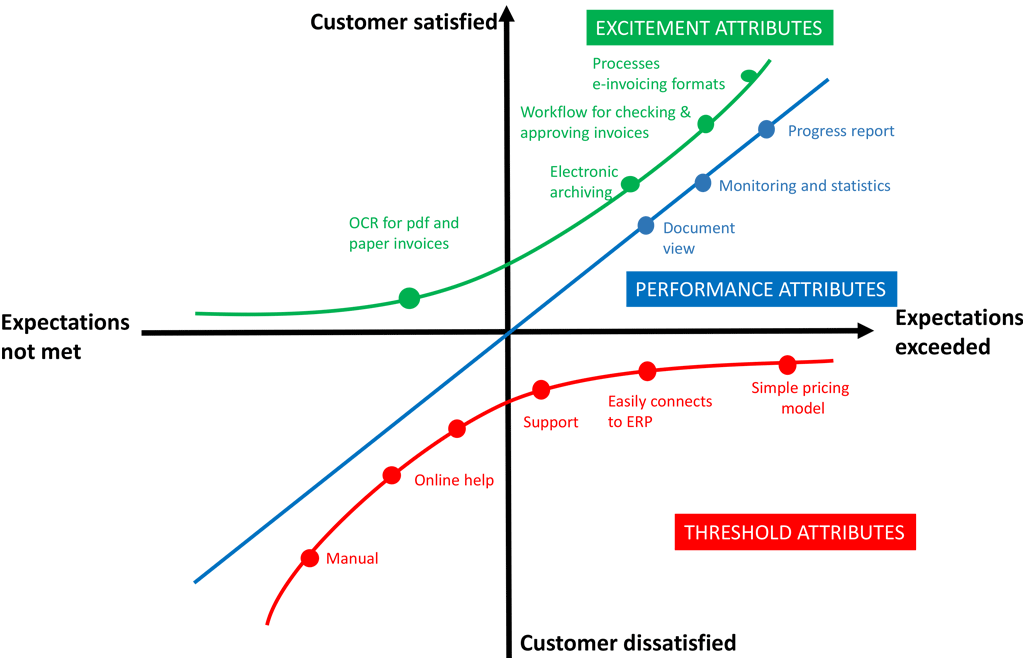
* Today's Delighters often become tomorrow's Satisfiers
* Satisfiers can eventually become Dissatisfiers as they become standard industry features

This classification system helps our team prioritize development efforts to ensure we first meet basic expectations (Dissatisfiers), then focus on competitive performance features (Satisfiers), and finally incorporate innovative elements (Delighters) that will differentiate our Campus Ride-Sharing Platform with Parking System Integration.

2 Elicitation Execution and Findings

2.1 Categorized Requirements (Based on Kano)

Define the categories: Dissatisfiers, Satisfiers, Delighters   
Sort the requirements into those categories above then create the kano model  
  
example



3 Appendices

3.1 References

fsdfsddf