Software Requirements Specification

for

Xpendit

Prepared by Aiden Lammert, Matthew Prescott, Matthew Ackley, Durgin Sweet, Joshua Schaff

February 26, 2019

Table of Contents

Table of Contents ii

Revision History ii

1. Introduction 1

1.1 Purpose 1

1.2 Document Conventions 1

1.3 Intended Audience and Reading Suggestions 1

1.4 Product Scope 1

1.5 References 1

2. Overall Description 2

2.1 Product Perspective 2

2.2 Product Functions 2

2.3 User Classes and Characteristics 2

2.4 Operating Environment 2

2.5 Design and Implementation Constraints 2

2.6 User Documentation 2

2.7 Assumptions and Dependencies 3

3. External Interface Requirements 3

3.1 User Interfaces 3

3.2 Hardware Interfaces 3

3.3 Software Interfaces 3

3.4 Communications Interfaces 3

4. System Features 4

4.1 System Feature 1 4

4.2 System Feature 2 (and so on) 4

5. Other Nonfunctional Requirements 4

5.1 Performance Requirements 4

5.2 Safety Requirements 5

5.3 Security Requirements 5

5.4 Software Quality Attributes 5

5.5 Business Rules 5

6. Other Requirements 5

Appendix A: Glossary 5

Appendix B: Analysis Models 5

Appendix C: To Be Determined List 6

# Introduction

## Purpose

Xpendit is a group finance managing application designed to help friends and roommates keep track of what they owe each other. The application will allow groups of people to track any fee that a group of people might share with each other, such as rent, apartment maintenance, or the cost of going out to eat. Regularly recurring costs can be set up to ensure that continual expenses such as rent or utility bills are paid on time. Xpendit will also allow groups to create shared shopping lists to make it easy to keep track of what the group needs, how much each item costs, and how much each member of the group needs to contribute financially.

## Document Conventions

All requirements are listed within the document and with their own priority. High-level requirement are assumed to share priority level with lower level requirements.

## Intended Audience and Reading Suggestions

The Xpendit System Requirement document is intended to use by the project team. Management, developing, document writing, and testing will be divided among the five members of the team. The System Requirement Specifications document outlines the scope of the project and overview. More specifically, the document covers the design and implementation requirements for the UI and underlying system of the final product.

## Product Scope

Xpendit’s goal is provide a simple way for groups to tracks finances. Users create groups with other users to tracks expenses shared between each other extending from recurring bills to lending each other money. Users can create or join multiple groups while still tracking the total expenses to help manage their finances. Xpendit allows for flexibility in tracking any type of expenses and the amount shared between each member of a group.

## References

# Overall Description

## Product Perspective

Xpendit is a recreation of other similar roommate applications, but with more flexibility. Other applications only offered some of the features of Xpendit and were limited in certain aspects. This is not a replacement for another product, but its own self-contained product.

## Product Functions

* Individual user accounts
  + Creation of accounts
  + Managing Accounts
* Creating rooms for users
  + Tracks expenses between users of rooms
  + Provide a group shopping list
  + Provide a group bulletin board
  + Add users to room
  + Remove users from room
  + Delete room
* Track user expenses
  + Add expense
  + Remove expense
  + Show total of all expenses
  + Show who expenses are owed to
  + Allow users to dispute charges from other users
  + Allow expenses to be split among groups evenly, by percentage or specific dollar amounts
  + Manual payment confirmation of expenses
  + Prioritize expenses by age, amount due or custom priority

## User Classes and Characteristics

<Identify the various user classes that you anticipate will use this product. User classes may be differentiated based on frequency of use, subset of product functions used, technical expertise, security or privilege levels, educational level, or experience. Describe the pertinent characteristics of each user class. Certain requirements may pertain only to certain user classes. Distinguish the most important user classes for this product from those who are less important to satisfy.>

## Operating Environment

Xpendit will operate on Android 8.0 Oreo. The majority of Android devices currently operate on this version. The app is designed to be standalone and will no interact with any other systems.

## Design and Implementation Constraints

Flutter will be used for the development of the application. Development of the application will be restricted to Android, thus will be restricted to the hardware limitations of Android devices. The application will keep user’s data secure as well as not track user location.

## User Documentation

The app will be released without any external documentation.

## Assumptions and Dependencies

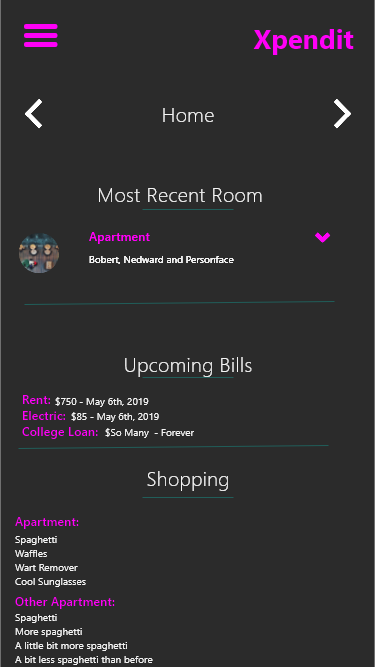
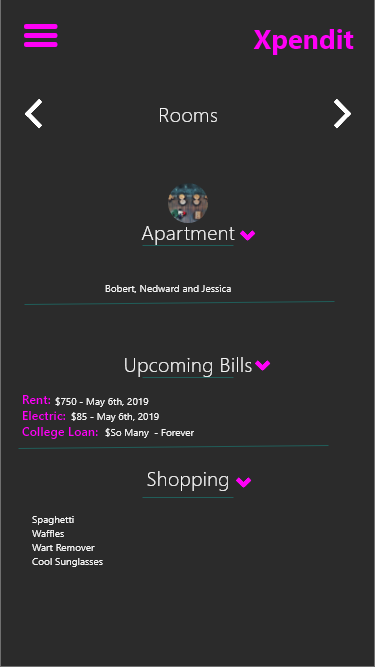
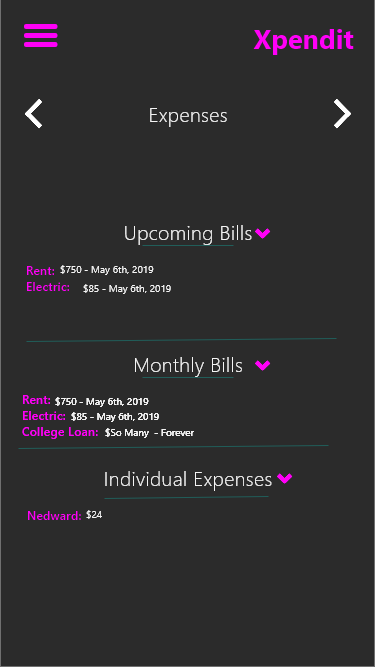
All requirements are expectation are based on having the required hardware to host the application from and

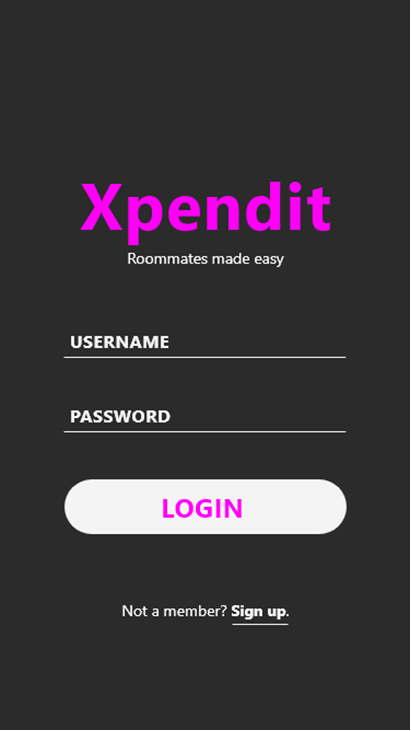
# External Interface Requirements

## User Interfaces

Xpendit is designed to use with a touch screen device. All interaction between the user the application will be through on-screen buttons and keyboards.

**Mock-up UI:**





## Hardware Interfaces

The application is built for Android device. It is meant to be standalone, except for having to interact with the account database.

## Software Interfaces

<Describe the connections between this product and other specific software components (name and version), including databases, operating systems, tools, libraries, and integrated commercial components. Identify the data items or messages coming into the system and going out and describe the purpose of each. Describe the services needed and the nature of communications. Refer to documents that describe detailed application programming interface protocols. Identify data that will be shared across software components. If the data sharing mechanism must be implemented in a specific way (for example, use of a global data area in a multitasking operating system), specify this as an implementation constraint.>

## Communications Interfaces

Xpendit applications will communicate with the user and group database server.

# System Features

<This template illustrates organizing the functional requirements for the product by system features, the major services provided by the product. You may prefer to organize this section by use case, mode of operation, user class, object class, functional hierarchy, or combinations of these, whatever makes the most logical sense for your product.>

## System Feature 1

<Don’t really say “System Feature 1.” State the feature name in just a few words.>

4.1.1 Description and Priority

<Provide a short description of the feature and indicate whether it is of High, Medium, or Low priority. You could also include specific priority component ratings, such as benefit, penalty, cost, and risk (each rated on a relative scale from a low of 1 to a high of 9).>

4.1.2 Stimulus/Response Sequences

<List the sequences of user actions and system responses that stimulate the behavior defined for this feature. These will correspond to the dialog elements associated with use cases.>

4.1.3 Functional Requirements

<Itemize the detailed functional requirements associated with this feature. These are the software capabilities that must be present in order for the user to carry out the services provided by the feature, or to execute the use case. Include how the product should respond to anticipated error conditions or invalid inputs. Requirements should be concise, complete, unambiguous, verifiable, and necessary. Use “TBD” as a placeholder to indicate when necessary information is not yet available.>

<Each requirement should be uniquely identified with a sequence number or a meaningful tag of some kind.>

REQ-1:

REQ-2:

## System Feature 2 (and so on)

# Other Nonfunctional Requirements

## Performance Requirements

<If there are performance requirements for the product under various circumstances, state them here and explain their rationale, to help the developers understand the intent and make suitable design choices. Specify the timing relationships for real time systems. Make such requirements as specific as possible. You may need to state performance requirements for individual functional requirements or features.>

## Safety Requirements

<Specify those requirements that are concerned with possible loss, damage, or harm that could result from the use of the product. Define any safeguards or actions that must be taken, as well as actions that must be prevented. Refer to any external policies or regulations that state safety issues that affect the product’s design or use. Define any safety certifications that must be satisfied.>

## Security Requirements

<Specify any requirements regarding security or privacy issues surrounding use of the product or protection of the data used or created by the product. Define any user identity authentication requirements. Refer to any external policies or regulations containing security issues that affect the product. Define any security or privacy certifications that must be satisfied.>

## Software Quality Attributes

<Specify any additional quality characteristics for the product that will be important to either the customers or the developers. Some to consider are: adaptability, availability, correctness, flexibility, interoperability, maintainability, portability, reliability, reusability, robustness, testability, and usability. Write these to be specific, quantitative, and verifiable when possible. At the least, clarify the relative preferences for various attributes, such as ease of use over ease of learning.>

## Business Rules

<List any operating principles about the product, such as which individuals or roles can perform which functions under specific circumstances. These are not functional requirements in themselves, but they may imply certain functional requirements to enforce the rules.>

# Other Requirements

<Define any other requirements not covered elsewhere in the SRS. This might include database requirements, internationalization requirements, legal requirements, reuse objectives for the project, and so on. Add any new sections that are pertinent to the project.>

Appendix A: Glossary

<Define all the terms necessary to properly interpret the SRS, including acronyms and abbreviations. You may wish to build a separate glossary that spans multiple projects or the entire organization, and just include terms specific to a single project in each SRS.>

Appendix B: Analysis Models

<Optionally, include any pertinent analysis models, such as data flow diagrams, class diagrams, state-transition diagrams, or entity-relationship diagrams.>

Appendix C: To Be Determined List

<Collect a numbered list of the TBD (to be determined) references that remain in the SRS so they can be tracked to closure.>