Software Requirements Specification

for

EasyPay

Version 1.0

Prepared by

Group Name: EasyPay

|  |  |  |
| --- | --- | --- |
| Ubaid Mukadam | 60004210229 | ubaidmukadam95@gmail.com |
| Raj Singh | 60004210187 | rajsingh10@gmail.com |
| Raj Ghag | 60004210231 | rajghag0904@gmail.com |

|  |  |
| --- | --- |
| Instructor: | Dr. Kiran Bhowmick |
| Course: | Software Engineering |
| Lab Section: | C31 |
| Date: | 14 – 02 - 2024 |

Contents

[1 Introduction 3](#__RefHeading___Toc750_2343461759)

[1.1 Document Purpose 3](#__RefHeading___Toc752_2343461759)

[1.2 Product Scope 3](#__RefHeading___Toc754_2343461759)

[1.3 Intended Audience and Document Overview 3](#__RefHeading___Toc756_2343461759)

[1.4 Definitions, Acronyms and Abbreviations 4](#__RefHeading___Toc758_2343461759)

[1.5 Document Conventions 4](#__RefHeading___Toc760_2343461759)

[1.6 References and Acknowledgments 4](#__RefHeading___Toc762_2343461759)

[2 Overall Description 5](#__RefHeading___Toc764_2343461759)

[2.1 Product Perspective 5](#__RefHeading___Toc766_2343461759)

[2.2 Product Functionality 5](#__RefHeading___Toc768_2343461759)

[2.3 Users and Characteristics 5](#__RefHeading___Toc770_2343461759)

[2.4 Operating Environment 6](#__RefHeading___Toc772_2343461759)

[2.5 Design and Implementation Constraints 6](#__RefHeading___Toc774_2343461759)

[2.6 User Documentation 6](#__RefHeading___Toc776_2343461759)

[2.7 Assumptions and Dependencies 6](#__RefHeading___Toc778_2343461759)

[3 Specific Requirements 7](#__RefHeading___Toc780_2343461759)

[3.1 External Interface Requirements 7](#__RefHeading___Toc782_2343461759)

[3.2 Functional Requirements 9](#__RefHeading___Toc784_2343461759)

[3.3 Behaviour Requirements 10](#__RefHeading___Toc786_2343461759)

[4 Other Non-functional Requirements 12](#__RefHeading___Toc788_2343461759)

[4.1 Performance Requirements 12](#__RefHeading___Toc790_2343461759)

[4.2 Safety and Security Requirements 12](#__RefHeading___Toc792_2343461759)

[4.3 Software Quality Attributes 12](#__RefHeading___Toc794_2343461759)

Revisions

| Version | Primary Author(s) | Description of Version | Date Completed |
| --- | --- | --- | --- |
| 1.0 | Ubaid Mukadam  Raj Singh  Raj Ghag | Initial Version of SRS | 21/02/24 |

# 

# 

# Introduction

EasyPay is a user-friendly mobile application that simplifies everyday financial transactions. It offers a secure platform for mobile recharges, utility bill payments, and peer-to-peer transactions. The app uses cutting-edge technology for contactless payments, QR code transactions, and instant fund transfers. It also includes features for online shopping, travel bookings, and investment options. EasyPay's intuitive interface and stringent security measures make it a reliable solution for modern financial interactions.

## Document Purpose

This Software Requirements Specification (SRS) document pertains to the software system for EasyPay, a mobile application aimed at revolutionizing digital financial transactions. The purpose of this document is to outline the requirements and scope of the EasyPay application. It specifies the features and functionalities of the software, its intended audience, and the organization of the document. This SRS serves as a guide for developers, project managers, testers, and other stakeholders involved in the development and deployment of EasyPay.

In addition to outlining the features and functionalities, this document will also address the performance requirements, security considerations, and regulatory compliance standards that the EasyPay application must adhere to. By providing a comprehensive overview of the project's goals and constraints, this SRS aims to ensure alignment among stakeholders and facilitate the successful development and delivery of EasyPay.

## Product Scope

EasyPay is envisioned as a robust and user-friendly mobile application designed to streamline and enhance everyday financial experiences. It offers a comprehensive suite of features, including mobile recharges, utility bill payments, peer-to-peer transactions, online shopping, travel bookings, and investment options. The primary objective is to provide a seamless and secure platform for a wide array of financial activities, catering to both individual consumers and businesses. By integrating cutting-edge technology and stringent security measures, EasyPay aims to redefine digital transactions and position itself as a reliable and convenient solution in the evolving landscape of digital finance.

## Intended Audience and Document Overview

The intended audience for this document includes developers, project managers, testers, and other stakeholders involved in the development and deployment of EasyPay. It provides a comprehensive overview of the software requirements, organized into sections for easy reference. The document begins with an overview of the purpose and scope, followed by specific requirements related to external interfaces, functionality, users, operating environment, design constraints, user documentation, assumptions, and dependencies. A suggested sequence for reading the document is provided, starting with the overview sections and proceeding to the sections most pertinent to each reader type.

## Definitions, Acronyms and Abbreviations

Abbreviations and acronyms used in this document:

* SRS: Software Requirements Specification
* UPI: Unified Payments Interface
* GUI: Graphical User Interface
* FTP: File Transfer Protocol
* HTTP: Hypertext Transfer Protocol

## Document Conventions

This document follows the IEEE formatting requirements, utilizing Arial font size 11 throughout. Text is single-spaced with 1” margins. Section and subsection titles adhere to the provided template. Formatting conventions are standardized to ensure consistency and clarity.

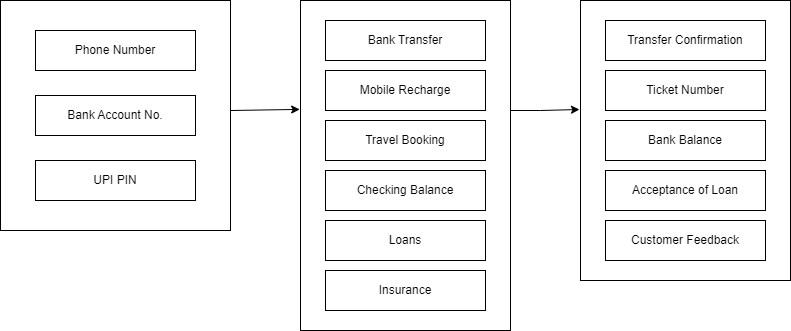
## References and Acknowledgments

Our app EasyPay takes inspiration from various different applications in the market like Google Pay, PayTM, PhonePe, etc.

# Overall Description

## Product Perspective

EasyPay is a standalone mobile application designed to operate within the digital financial ecosystem. It interfaces with various external systems such as payment gateways, banking networks, and merchant platforms to facilitate seamless transactions. The application's architecture comprises client-side components for user interaction and server-side components for data processing and storage.



## Product Functionality

* Mobile recharges
* Utility bill payments
* Peer-to-peer transactions
* Contactless payments
* QR code transactions
* Instant fund transfers through UPI
* Online shopping
* Travel bookings
* Investment options

## Users and Characteristics

EasyPay targets a diverse user base, including individual consumers, businesses, and tech-savvy users. The application caters to varying levels of technical expertise, frequency of use, and security considerations. Recognizing these characteristics is vital for tailoring the interface, features, and security measures to meet the specific needs of different user segments. User segmentation guides the design and development process, ensuring a personalized and satisfying experience for each category of users interacting with EasyPay.

## Operating Environment

EasyPay operates within a dynamic and versatile operating environment, designed for smartphones and tablets running on the latest versions of Android (7.0 and above) and iOS (10.0 and above). The hardware requirements include a camera for QR code scanning, internet connectivity, and sufficient memory and processing power. The application interfaces with external entities such as financial institutions and merchants, relying on APIs and protocols for secure communication.

The software components of EasyPay include the application itself, integrating with the mobile operating system, and external elements like libraries for QR code scanning and encryption protocols for secure transactions. Regular updates ensure compatibility with new releases and optimal performance. A system diagram visually represents the major components of EasyPay's operating environment, highlighting interactions and relationships. This understanding of the operating environment ensures the application's seamless functionality and adaptation to evolving technologies.

## Design and Implementation Constraints

EasyPay operates within specified design and implementation constraints that guide its development and deployment. These constraints encompass hardware limitations, interfaces with external applications such as the Unified Payments Interface (UPI), stringent security considerations, adherence to operating system requirements (Android and iOS), and compliance with design conventions and programming standards. Addressing these constraints is essential to ensure the application's efficiency, security, and seamless integration with various devices and platforms, contributing to EasyPay's success as a reliable and user-friendly digital financial platform.

## User Documentation

EasyPay prioritizes user documentation to enhance the user experience. This includes detailed user manuals offering step-by-step instructions for various functionalities, on-screen help integrated within the app for real-time guidance, and interactive tutorials to facilitate hands-on learning. The documentation aims to provide users with accessible and up-to-date information, ensuring they can navigate and utilize EasyPay's features effectively.

## Assumptions and Dependencies

Understanding assumptions and dependencies is vital for anticipating challenges during EasyPay's development and deployment. Assumptions include user access to smartphones meeting minimum requirements, stable internet connections, and the UPI infrastructure's stability. Dependencies encompass reliance on hardware and network providers, third-party services like UPI, and adherence to security standards and regulations. Acknowledging these aspects allows for proactive measures and risk mitigation throughout the application's lifecycle.

# Specific Requirements

## External Interface Requirements

### User Interfaces

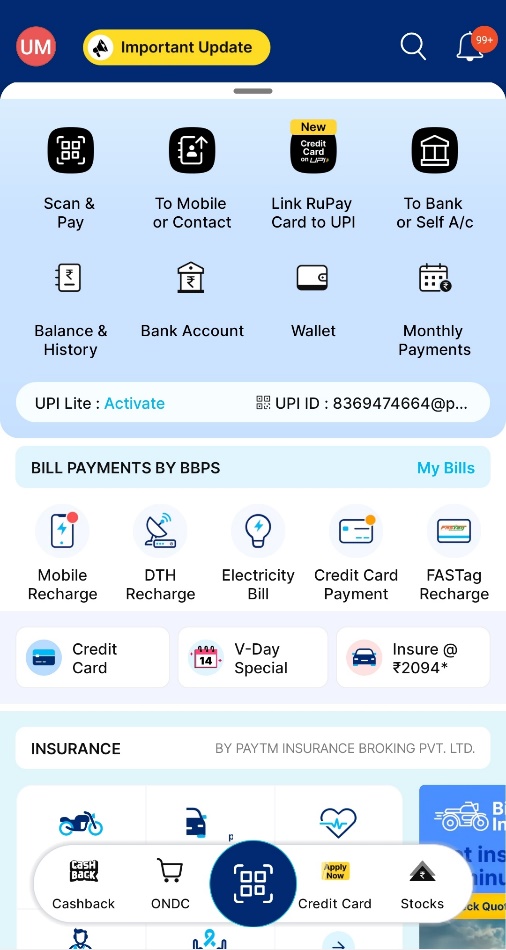
EasyPay features a user-friendly GUI with intuitive navigation and clear visual cues. Screens include:

**Login/Registration:**

This is the page which appears when the app is initially opened for the very first time.

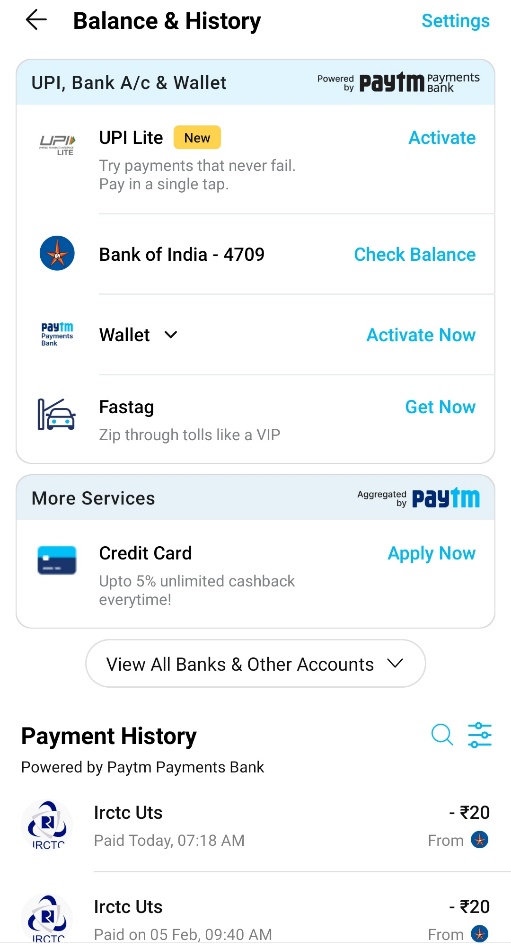
**Dashboard:**

The dashboard consists of various features which can be seen in the image below.



**Transaction History and Balance:**

This is the page which allows the user to see their bank balance and transaction history.



**Additional features:**

This app consists of various features in addition to payments such as ticket booking, insurance, buying digital gold, etc.

### Software Interfaces

For the EasyPay application, the primary software interface is with the operating system of mobile devices. EasyPay will be compatible with both Android and iOS operating systems, including their respective versions such as Android 11 and iOS 15. The application will utilize platform-specific APIs and frameworks provided by Google and Apple for seamless integration with the underlying operating system. This interface enables EasyPay to access device hardware features such as camera, GPS, and network connectivity required for functionalities like scanning QR codes, detecting location for transactions, and processing payments securely. Additionally, the app will utilize native user interface components and behaviors consistent with the respective operating system's design guidelines, ensuring a familiar and intuitive experience for users on Android and iOS devices.

### Communications Interfaces

For EasyPay, communication functions are crucial for enabling seamless transactions between users, merchants, and financial institutions. The application will utilize secure communication protocols such as HTTPS for interactions with the backend servers, ensuring data integrity and confidentiality during transmission. Additionally, EasyPay will support various communication channels, including email notifications for transaction confirmations and account updates, SMS notifications for authentication and alerts, and push notifications for real-time updates on payment statuses and promotions.

Message formatting will adhere to industry-standard formats such as JSON or XML for transmitting data between the client and server components. Encryption standards such as TLS will be employed to secure data in transit, safeguarding sensitive information like payment details and personal identifiers. Moreover, data transfer rates will be optimized to provide efficient and responsive communication, while synchronization mechanisms will ensure consistency across multiple devices and platforms, allowing users to access their accounts and transaction history seamlessly.

## Functional Requirements

**Functional Requirements:**

**1. Financial Transactions:**

* Mobile Recharges: Users should be able to recharge their mobile phones using EasyPay by selecting their service provider, entering the mobile number, and specifying the recharge amount.
* Utility Bill Payments: EasyPay should allow users to pay utility bills such as electricity, water, gas, and internet bills by selecting the respective service provider, entering the account details, and specifying the payment amount.
* Peer-to-Peer Transactions: Users should be able to transfer funds to other EasyPay users by specifying the recipient's phone number or email address, along with the transfer amount.
* Contactless Payments: EasyPay should support contactless payments at participating merchants using NFC-enabled devices or QR codes for secure and convenient transactions.
* QR Code Transactions: Users should be able to scan QR codes displayed by merchants to make payments for goods and services directly from their EasyPay wallet.
* Instant Fund Transfers through UPI: EasyPay should integrate with the Unified Payments Interface (UPI) to facilitate instant fund transfers between bank accounts using UPI IDs or mobile numbers.

**2. E-commerce Integration:**

* Online Shopping: EasyPay should enable users to make purchases from e-commerce platforms by securely linking their EasyPay wallet to their shopping accounts and completing transactions seamlessly.

**3. Travel and Booking Services:**

* Travel Bookings: Users should be able to book flights, hotels, and other travel-related services through EasyPay by selecting their desired options, providing necessary details, and completing the payment process securely.

**4. Investment Options:**

* Investment Opportunities: EasyPay should provide users with access to various investment options such as mutual funds, stocks, and bonds, allowing them to invest directly from their EasyPay accounts and track their investments conveniently.

These functional requirements define the core operations and capabilities of the EasyPay application, covering a wide range of financial transactions, e-commerce integrations, travel bookings, and investment services to meet the diverse needs of users.

## Behaviour Requirements

### Use Case View

**Actors:**

1. Customer
2. Manager

**Use Cases:**

**Add Account:** The customer creates a new account with the bank.

**Check Balance:** The customer views their account balance.

**Credit:** The customer deposits money into their account.

**Debit:** The customer withdraws money from their account.

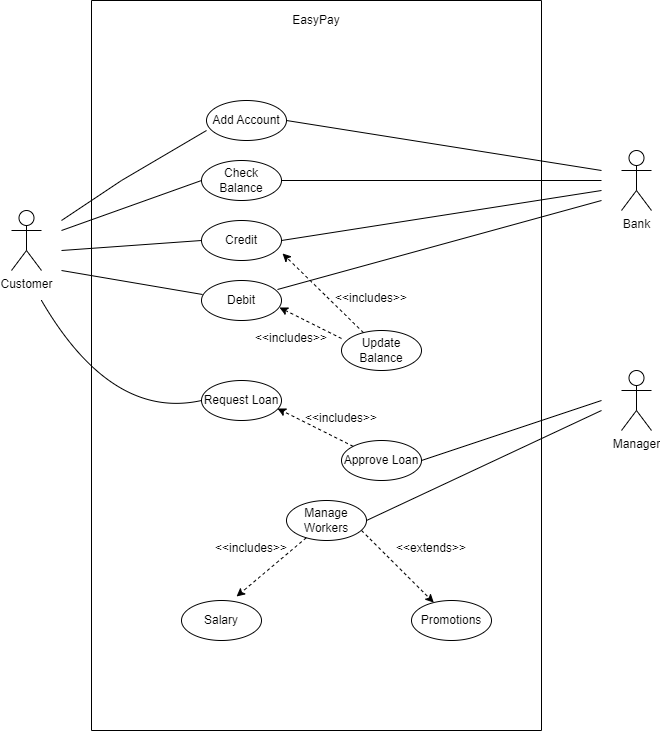
**Request Loan:** The customer applies for a loan from the bank.

**Approve Loan:** The manager approves or denies the customer's loan request.

**Manage Workers:** The manager manages the bank's employees.

**Salary:** The bank pays salaries to its employees.

**Promotions:** The manager promotes employees within the bank.



# Other Non-functional Requirements

## Performance Requirements

* Any transaction, including mobile recharges, bill payments, and fund transfers, should be completed within 10 seconds to ensure a seamless user experience.
* The app should handle a minimum of 10,000 simultaneous users without significant performance degradation to accommodate its dynamic user base.
* Response time for accessing online shopping and travel booking features should be under 3 seconds to enhance user satisfaction.
* The app should be able to process a minimum of 100 transactions per second during peak usage hours to maintain efficiency.
* The app should have a system uptime of at least 99.9% to ensure continuous availability for users.

## Safety and Security Requirements

* In case of a failed transaction, the app should automatically initiate a refund process within 24 hours to prevent financial loss for the user.
* The app must have a two-factor authentication (2FA) system for high-value transactions to ensure secure user identity verification.
* The app should comply with industry-standard encryption protocols (e.g., TLS 1.3) to protect user data during both transit and storage.
* Regular security audits and vulnerability assessments should be conducted every six months to identify and address potential threats.
* The app should adhere to local and international data protection regulations, such as GDPR, to safeguard user privacy and data.

## Software Quality Attributes

* **Reliability**

1. The app should have a system for automatic data backup every 24 hours to prevent data loss in case of system failures.
2. A comprehensive error logging system should be in place to track and analyze any system errors, ensuring quick issue resolution.

* **Usability**

1. The app's user interface should be designed with a focus on simplicity, ensuring that users can perform basic transactions without extensive training.
2. Regular user feedback surveys should be conducted, and the results should be used to make continuous improvements to the app's usability.

Appendix B - Group Log

|  |  |  |
| --- | --- | --- |
| **Date** | **Actors** | **Work Done** |
| 07/02/2024 | Ubaid Mukadam | Analyzed Requirements |
| 07/02/2024 | Raj Singh | Analyzed Requirements |
| 07/02/2024 | Raj Ghag | Analyzed Requirements |
| 14/02/2024 | Ubaid Mukadam | Prepared SRS |
| 14/02/2024 | Raj Singh | Prepared SRS |
| 14/02/2024 | Raj Ghag | Prepared SRS |
| 21/02/2024 | Ubaid Mukadam | Prepared SRS |
| 21/02/2024 | Raj Singh | Prepared SRS |
| 21/02/2024 | Raj Ghag | Prepared SRS |