# Software Requirements Specification (SRS) Document
# Project Name – Finease
# Team Number – 2

## Team Members -

**Swam Singla**

**Kushagra Trivedi**

**Ronak Gaur**

**Ayush Gupta**

**Nidhi Vaidya**

### Brief problem statement

*Finease is a comprehensive platform that streamlines business operations such as data entry, reconciliation, and regulatory compliance for ITR, GST, and PF filings. The platform leverages WhatsApp and a custom chatbot for user interaction, automates document processing, and provides functionalities for generating E-Waybills, delivery challans, and financial reports.*

### System requirements

**FinEase Project Technology Stack:**

- **Frontend & Backend:**
  - Node.js
  - Express.js
  - React.js (Web App)
  - MongoDB
- **APIs & Integrations:**
  - Gemini
  - NanoNets
- **Storage:**
  - Structured database for operational and compliance data (MongoDB Atlas)
- **Chatbot Integration:**
  - WhatsApp API
  - Custom app chatbot
- **Document Processing:**
  - File uploads for invoices, Form 16, and challans

**Tools and Libraries:**

- **Python Libraries:**

    - py-tesseract for OCR
    - PIL (Python Imaging Library) for image processing
- **Frontend Styling:**
    - TailwindCSS
- **Tools:**
    - Postman (API Testing)
    - Ngrok (Local hosting)
    - Twilio (Webhook)
    - NanoNets (Model Training)
- **Collaboration Tools:**
    - GitHub for version control

**Deployment:**

- Vercel (Frontend)
- Render (Backend)
- Docker (Backend)

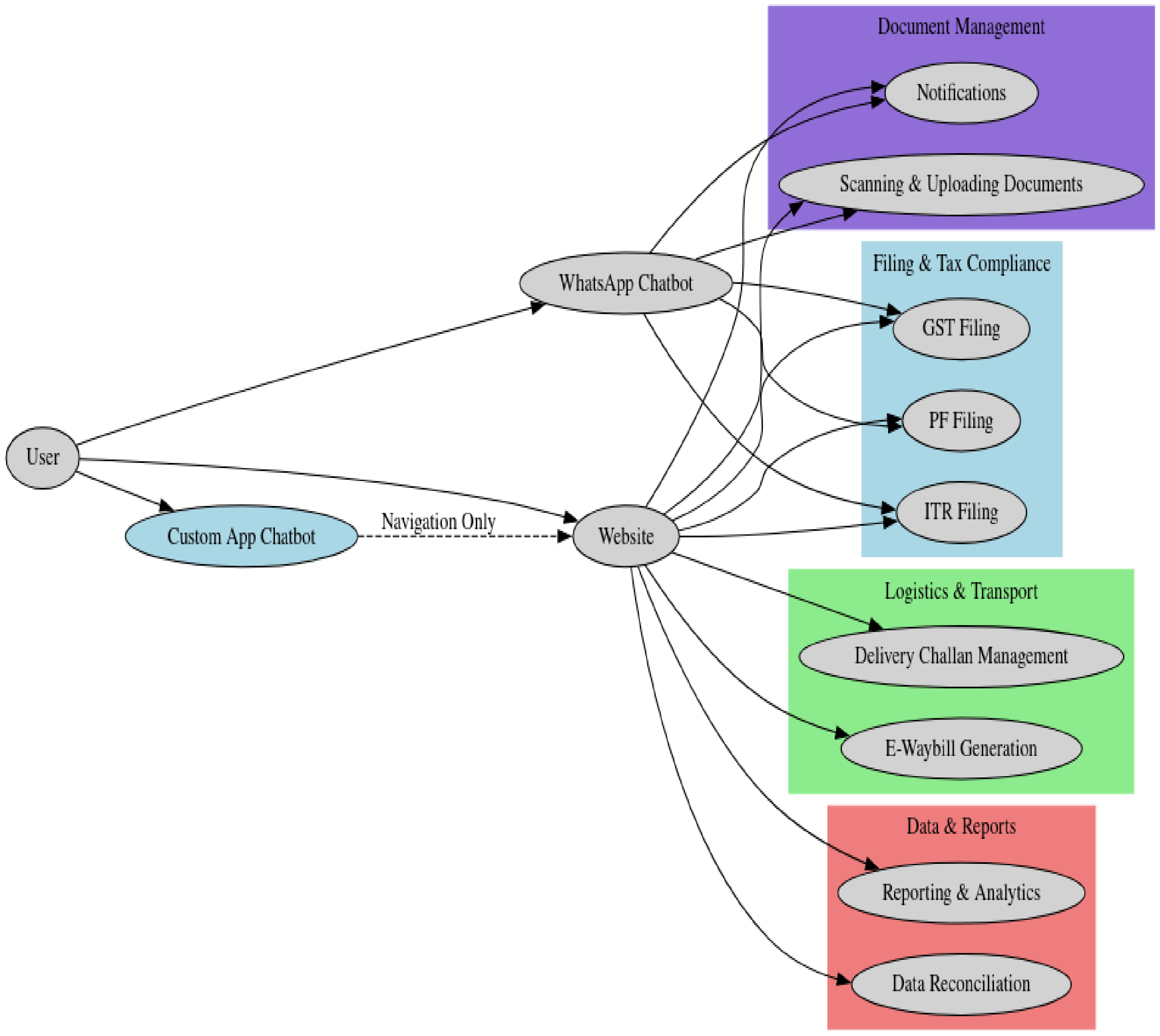
## User Profiles

1. **Business Owners & Accountants**
    - **Familiarity Level:** Basic-to-Medium
    - **Role:** Utilize automation for tax filings, invoice generation, and reconciliation. Reduced manual effort due to AI-powered processing.
2. **Employees & HR Personnel**
    - **Familiarity Level:** Basic
    - **Role:** Submit financial records and compliance documents. Now also interact with **document classification and label extraction** features.
3. **Tax Professionals**
    - **Familiarity Level:** Advanced
    - **Role:** Leverage **automated insights, analytics, and compliance reporting** to streamline filings instead of manual data processing.

## Feature requirements (described using use cases)

| No. | User Case Name | Description | Release |
|---|---|---|---|
| 1. | Whatsapp Integration | Collects invoices, purchase records from users via WhatsApp | R1 |
| 2. | Custom App Chatbot | Guides users through the website for seamless navigation. | R1 |
| 3. | E-Waybill Generation | Automates E-Waybill creation through API integration | R1 |
| 4. | Delivery Challan Management | Generates challans for goods transport | R1 |
| 5. | ITR Filing | Automates document submission by extracting data from Form 16 to pre-fill ITR forms through secure government API integration. Supports testing in a risk-free sandbox environment. | R2 |
| 6. | GST Filing | Automates GSTR-1 and GSTR-3B submissions through secure government API integration. Enables testing in a sandbox environment. | R2 |
| 7. | PF Filing | Generates ECR files for the EPFO portal using secure government API integration. Supports sandbox testing for safe API interactions. | R2 |
| 8. | Data Reconciliation | Identifies mismatched invoices and records | R2 |
| 9. | Reporting,storage & Analytics | Generates summaries and exports reports in PDF/Excel, can also store these summaries | R2 |
| 10. | Scanning and uploading Documents | Scan and upload documents for GST//PF/ITR filing and classify them and extract data from them using external API | R1 |
| 11. | Notifications | Sending automated tax and document reminders via WhatsApp messages and ensuring timely notifications for deadlines and important updates. | R2 |

## Use case diagram



## Use case description

**Note-** **<u>App here corresponds to Webapp and not mobile app</u>**

**Use Case Number:** UC-01

**Use Case Name:** GST Filing

**Overview:**
This use case describes how users can file **GSTR-1 and GSTR-3B** returns through secure government **API integration**. The system allows users to test filings in a **sandbox environment** before making actual submissions.

### Actors:

- **User** (Business Owner/Accountant)
- **Website**
- **Backend System**
- **GSTN API**

**Pre-conditions:**

- The user must have a valid **GSTIN**.
- Sales and purchase invoices must be recorded in the system.
- The system must be integrated with the **GSTN API**.

### Flow:

### Main (Success) Flow:

1. The user logs into the **website** and navigates to the **GST Filing** section.
2. The system retrieves relevant sales and purchase data.
3. The system **validates** GSTIN, HSN codes, and tax amounts.
4. The **GSTR-1 or GSTR-3B** form is **auto-filled** based on the available data.
5. The user **reviews** the filing details and confirms submission.
6. The system submits the return through the **GSTN API**.
7. The user receives a confirmation message and a filing status update.

**Alternate Flows:**

- **Invalid Data:** If incorrect **GSTIN, HSN codes, or tax amounts** are detected, the system prompts the user for corrections.
- **Sandbox Testing:** The user can opt for a **test submission** in a sandbox environment before filing officially.
- **Submission Failure:** If the API fails, the system notifies the user and provides options for retrying or manual submission.

### Post-condition:

- The **GST return is successfully submitted** via API.
- A filing record is **stored** in the system for future reference.
- The user receives a **confirmation message** upon successful submission.

**Use Case Number:** UC-02

**Use Case Name:** WhatsApp Integration

**Overview:**
This use case describes the process of integrating WhatsApp as a platform for user interaction, allowing users to submit invoices, purchase records, and other financial data, request filings, and receive notifications about their tax and compliance activities.

**Actors:**

- User (Business Owner/Accountant)
- WhatsApp Chatbot
- Backend System

**Pre-condition:**

- The user must have an active WhatsApp account.
- The chatbot must be authorized and integrated with the system.
- Required APIs (WhatsApp Business API, backend processing) must be configured.

**Flow:**

**Main (Success) Flow:**

1. The user initiates a conversation with the WhatsApp chatbot.
2. The chatbot presents available options (e.g., GST filing, PF filing, ITR filing, document classification).
3. The user selects an option and submits relevant data (e.g., invoice details, Form 16 upload).
4. The chatbot validates the data and stores it in the backend.
5. If necessary, the chatbot requests additional information or corrections.
6. The system processes the request (e.g., prepares GST filing).
7. The user receives a confirmation message with details and next steps.

**Alternate Flows:**

- **Invalid Data:** If the submitted data is incomplete or incorrect, the chatbot prompts the user to provide corrections.

**Post Condition:**

- The user’s request is successfully processed, and the relevant data is stored.

**Use Case Number:** UC-03

**Use Case Name:** Custom App Chatbot

## Overview:

This use case describes the functionality of a custom app chatbot integrated within the website. The chatbot assists users by providing basic guidance on how to file GST, ITR, and PF filings, as well as how to generate invoices, E-Waybills, and delivery challans on the website. It does not collect or validate data but helps users navigate the relevant sections of the website.

### Actors:

- **User (Business Owner/Accountant)**
- **Custom App Chatbot**
- **Website**

### Pre-condition:

- The user must have access to the website where the chatbot is integrated.
- The chatbot must be active and capable of providing responses based on user queries.

### Flow:

## Main (Success) Flow:

1. The user accesses the website and initiates a conversation with the chatbot.
2. The user asks a question related to filing **GST, ITR, PF**, or generating **invoices/E-Waybills/delivery challans**.
3. Based on the user's query, the chatbot provides relevant guidance in text format (e.g., steps to follow, required documents).
4. The user follows the chatbot's guidance and completes the process within the website.

**Alternate Flows:**

- **User Asks an Unsupported Question:** If the chatbot does not recognize the question, it responds with alternative suggestions such as:
    - **"I'm sorry, but I don't have an answer to that. Could you please ask something else?"**

### Post Condition:

- The user is successfully guided to the relevant section within the website for filing or document generation.
- The chatbot ensures the user understands how to proceed but does not directly handle the filing process.

**Use Case Number:** UC-04

**Use Case Name:** E-Way Bill Generation

### Overview:

The **E-Way Bill Generation** use case describes how users can generate an E-Way Bill through the **website**. The system provides a structured interface where users can input relevant details. Once submitted, the system processes the request and provides the generated E-Way Bill for download.

### Actors:

- **User** (Business Owner/Transporter/Accountant)
- **Website**

**Pre-conditions:**

- The user must have access to the website.
- The required details (e.g., invoice details, consignee/consignor information, HSN codes) must be available.

### Flow:

### Main (Success) Flow:

1. The user accesses the website and navigates to the **E-Way Bill Generation** section.
2. The system prompts the user to enter details such as consignee/consignor details, HSN codes, invoice reference, and transport details.
3. The user submits the required information.
4. Upon successfully submitting, the system generates the E-Way Bill.
5. The E-way Bill gets downloaded.

**Alternate Flows:**

**NONE**

### Post-condition:

- The E-Way Bill is successfully generated and gets downloaded.
- The user receives confirmation via the website.

**Use Case Number:** UC-05

**Use Case Name:** Delivery Challan Management

**Overview:**
This use case describes the process of generating and managing delivery challans for various scenarios such as goods transportation, returns, or samples. The system allows users to create, store and retrieve challans for compliance and reconciliation purposes.

**Actors:**

- User (Business Owner/Transporter/Accountant)
- WhatsApp Chatbot
- Backend System

**Pre-condition:**

- The user must have shipment details, including invoice references, quantity, and consignee/consignor details.
- The system must have the ability to generate and store delivery challans.

**Flow:**

**Main (Success) Flow:**

1. The user initiates a request to generate a delivery challan via the app interface.
2. The system prompts the user to enter the required details, including:
    - Goods description and quantity
    - Shipment details (consignee/consignor information)
3. A delivery challan is generated and stored in the system.
4. The system provides the user with a downloadable and shareable PDF of the delivery challan.
5. The user receives a confirmation message via website.

**Alternate Flows:**

- **Challan Retrieval:** Users can retrieve previously generated challans for reconciliation and reporting.

**Post Condition:**

- The delivery challan is successfully generated and stored in the system.
- The user receives a confirmation along with a downloadable document.

## Use Case Number: UC-06

**Use Case Name:** Income Tax Return (ITR) Filing

### Overview:

This use case describes how users can file **Income Tax Returns (ITR)** by extracting data from **Form 16 and other financial documents** through secure government **API integration**. The system allows users to **test filings in a sandbox environment** before making actual submissions.

### Actors:

- **User** (Taxpayer/Accountant/Business Owner)
- **Website**
- **Backend System**
- **IT Portal API**

**Pre-conditions:**

- The user must have valid financial documents, including **Form 16, salary details, and tax deduction records**.
- The system must be integrated with the **IT Portal API** (if applicable).
- The system must support **data extraction and validation** for ITR filing.

### Flow:

### Main (Success) Flow:

1. The user logs into the **website** and navigates to the **ITR Filing** section.
2. The system prompts the user to **upload Form 16** and other necessary financial documents.
3. The system extracts relevant data and **pre-fills the ITR form**.
4. The system **validates** the extracted information, checking for errors or missing details.
5. The user **reviews** the filing details,can also manually change the details and confirms submission.
6. If API integration is available, the system **submits the ITR** to the **IT Portal** and retrieves a confirmation.
7. If manual submission is required, the system provides the user with a **submission-ready ITR file**.
8. The user receives a **confirmation message** with the filing status.

**Alternate Flows:**

- **Invalid Data:** If missing or incorrect **income, deductions, or tax details** are detected, the system prompts the user for corrections.
- **Sandbox Testing:** The user can opt for a **test submission** in a sandbox environment before filing officially.
- **API Failure:** If the IT Portal API is unavailable, the system notifies the user and provides a **submission-ready file** for manual upload.

### Post-condition:

- The **ITR is successfully filed** via API, or a **submission-ready file** is generated.
- A filing record is **stored** in the system for future reference.
- The user receives a **confirmation message** upon successful submission.

## Use Case Number: UC-07

**Use Case Name:** Provident Fund (PF) Filing

### Overview:

This use case describes how users can file **Provident Fund (PF) contributions** by preparing and submitting **Employee Contribution Returns (ECR) files** through **secure government API integration**. The system allows users to **test filings in a sandbox environment** before making actual submissions.

### Actors:

- **User** (HR/Employer/Accountant)
- **Website**
- **Backend System**
- **EPFO Portal API**

**Pre-conditions:**

- The user must have valid **employee salary and PF contribution details**.
- The system must be capable of generating **ECR files** in the required format.
- The system must be integrated with the **EPFO Portal API**.

### Flow:

### Main (Success) Flow:

1. The user logs into the **website** and navigates to the **PF Filing** section.
2. The system prompts the user to enter or upload **salary details and employee PF contributions**.
3. The system **validates** the data and calculates **PF contributions** accordingly.
4. The system generates an **ECR file** in the required format.
5. The user **reviews** the details,can  manually change the details and confirms submission.
6. If API integration is available, the system **submits the ECR file** to the **EPFO portal** and retrieves a confirmation.
7. If manual submission is required, the system provides the user with a **downloadable ECR file** for manual upload.
8. The user receives a **confirmation message** with the filing status.

**Alternate Flows:**

- **Invalid Data:** If missing or incorrect **salary, PF contribution, or employee details** are detected, the system prompts the user for corrections.
- **Sandbox Testing:** The user can opt for a **test submission** in a sandbox environment before filing officially.
- **API Failure:** If the **EPFO portal API** is unavailable, the system notifies the user and provides the **ECR file** for manual upload.

### Post-condition:

- The **PF contributions are successfully filed** via API, or a **submission-ready ECR file** is generated.
- A filing record is **stored** in the system for future reference.
- The user receives a **confirmation message** upon successful submission.

## **Use Case Number:** UC-08

**Use Case Name:** Data Reconciliation

**Overview:**
This use case describes the process of comparing sales, purchases, and payments against recorded invoices to identify mismatches or missing data. The system alerts users to discrepancies and suggests corrective actions to ensure accurate financial reporting and regulatory compliance.

**Actors:**

- User (Business Owner/Accountant)
- WhatsApp Chatbot
- Backend System

**Pre-condition:**

- Sales, purchase, and payment records must be available in the system.
- The system should have access to previous invoice data for comparison.
- The user must have permission to access and correct reconciliation data.

**Flow:**

**Main (Success) Flow:**

1. The user initiates a data reconciliation request via the app interface.
2. The system retrieves sales, purchase, and payment records from the database.
3. The system compares transactions with existing invoices and identifies mismatches or missing data.
4. If discrepancies are found, the system generates a report highlighting inconsistencies.
5. The user receives a notification via app about the discrepancies.
6. The system suggests corrective actions, such as updating missing invoices or verifying payments.
7. The user reviews and approves the suggested corrections or manually updates records.
8. The system updates reconciled data and marks discrepancies as resolved.
9. The user receives a confirmation message upon successful reconciliation.

**Alternate Flows:**

- **No Discrepancies Found:** If all records match, the system confirms successful reconciliation.
- **User Opts for Manual Review:** If the user wants to manually review discrepancies, the system allows access to detailed reports.
- **Data Entry Errors:** If the system detects potential data entry errors, it prompts the user for corrections before final reconciliation.

**Post Condition:**

- The financial records are successfully reconciled, ensuring accuracy in tax filings and compliance reports.
- The user receives a confirmation along with a final reconciliation report.

## Use Case Number: UC-09

**Use Case Name:** Reporting and Analytics

**Overview:**
This use case describes the process of generating financial and compliance reports based on collected data. The system compiles summaries for GST filings, ITR submissions, and PF contributions, and other business transactions. Users can export reports in PDF or Excel formats for offline use.

**Actors:**

- User (Business Owner/Accountant)
- WhatsApp Chatbot
- Backend System

**Pre-condition:**

- The system must have access to structured financial and compliance data.
- The user must have appropriate permissions to generate reports.
- The backend system should support data aggregation and report formatting.

**Flow**
**Main (Success) Flow:**

1. The user initiates a report request via app interface.
2. The system retrieves relevant data from sales, purchases, tax filings, PF/ESI contributions, and other financial records.
3. The system processes the data and generates reports based on predefined formats.
4. The user is presented with options to select a report type (e.g., GST summary, ITR filing report, reconciliation report).
5. The system generates the requested report in real-time and presents a preview.
6. The user chooses a preferred format (PDF/Excel) and downloads the report.
7. The system sends a confirmation notification via app.

**Alternate Flows:**

- **Data Unavailable:** If required data is missing, the system notifies the user and suggests corrective actions.
- **User Customization:** The user can filter and customize report parameters before generation.

**Post Condition:**

- The requested report is successfully generated and made available for download.
- The user receives a confirmation along with the report file.

## **Use Case Number:** UC-10

**Use Case Name:** Scanning and Uploading Documents

**Overview:**
This use case describes the process of scanning and uploading financial and compliance-related documents, such as invoices, Form 16, challans, and other necessary paperwork. The system ensures proper categorization, storage, and retrieval of uploaded files for further processing.

**Actors:**

- User (Business Owner/Accountant)
- WhatsApp Chatbot
- Website
- Backend System

**Pre-condition:**

- The user must have documents ready for upload.
- The system must support document scanning, categorization, and storage.

**Flow**
**Main (Success) Flow:**

1. The user initiates a document upload request via WhatsApp chatbot or the app interface.
2. The system prompts the user to select the document type (e.g., invoice, Form 16, challan, tax receipt).
3. The user scans or selects an existing document and uploads it.
4. The system processes the document using OCR (if applicable) to extract relevant information.
5. The system validates the document format and checks for required fields using external API(Nanonets).
6. If the document is valid, it is stored in the appropriate category (e.g., compliance, transactions).
7. The user receives a confirmation message via WhatsApp or the app interface.

**Alternate Flows:**

- **Invalid Document Format:** If the uploaded document is in an unsupported format, the system prompts the user to retry with a valid format.
- **Failed Upload:** If the upload fails due to network issues, the system notifies the user and allows a retry.
- **Manual Data Entry:** If API processing fails, the system prompts the user to manually enter the extracted data.

**Post Condition:**

- The document is successfully uploaded and stored for future reference.
- The user receives a confirmation.

**Use Case Number:** UC-11
**Use Case Name:** Notifications
**Overview:**
This use case describes the process of sending automated notifications to users regarding important updates such as filing deadlines, document uploads via WhatsApp chatbot.

**Actors:**

- User (Business Owner/Accountant)
- WhatsApp Chatbot
- Backend System

**Pre-condition:**

- The user must have an active account with contact details registered.
- The system must have access to relevant compliance and financial data.

**Flow**
**Main (Success) Flow:**

1. The system detects an event that requires user notification (e.g., tax filing due date).
2. Based on predefined triggers, the system generates a notification message.
3. The system sends the notification via the preferred channel Whatsapp.
4. The user receives the notification and takes necessary action.
5. The system logs the notification status (sent, delivered, read).

**Alternate Flows:**

- **User Does Not Receive Notification:** If delivery fails, the system retries or notifies the user via an alternate channel.

**Post Condition:**

The user is informed of important updates related to financial and compliance processes.
The system ensures timely actions by sending alerts to prevent delays or penalties.