Cairo University
Faculty of Graduate Studies for Statistical Research

Software Requirements Specification “ SRS “

for

EASY INVOICING APPLICATION “EIA”

Version 1.0 approved

Prepared by 103 Group
Ahmed Gamal Sallam - 202201314
Kahlawy Saber Fathy - 202201347
Waled Ibrahim Mohey Eldin - 202202040
Ahmed Gomaa Shazly - 202201069
Mohamed Mohamed Selim - 202201271

Software Engineering Diploma

2022-2023

Table of Contents

Revision History

Name	Date	Reason For Changes	Version

1. Introduction

1.1 Purpose

The purpose of a Software Requirements Specification (SRS) document is to outline the necessary features, functionality, and constraints required for the development of a software system. The SRS document serves as a communication tool between the development team, stakeholders, and users, providing a clear understanding of the system requirements and design specifications.

The SRS document helps ensure that all stakeholders have a shared understanding of the system's functionality and that the development team has a clear set of requirements to follow during the software development process. The document also serves as a reference guide for testing and validating the system, ensuring that it meets all functional and non-functional requirements.

The SRS document typically includes sections outlining the system's purpose, proposed system features and capabilities, design specifications, testing criteria, and any other relevant information. The document is continuously updated throughout the software development lifecycle, providing a reference guide for the development team and stakeholders. Overall, the SRS document plays a critical role in ensuring the successful development of a software system that meets the needs and expectations of all stakeholders.

1.2 Intended Audience and Reading Suggestions

1) ***Intended Audience:***

Development Team: ***This includes software developers, programmers, and engineers who will be responsible for building and implementing the Easy Invoicing Application. Project Managers: Individuals responsible for overseeing the project, managing resources, and ensuring timely delivery.***

Quality Assurance Team: ***Testers and quality assurance professionals who will validate the application against the specified requirements.***

Business Analysts: ***Professionals who gather and analyze business requirements and help bridge the gap between business stakeholders and the development team.***

Client or Business Stakeholders: ***These are the individuals or organizations for whom the Easy Invoicing Application is being developed. They have a vested interest in the project's success.***

2) ***Reading Suggestions:***

Executive Summary: ***Provide a concise overview of the SRS, highlighting the project's objectives, scope, and key features. This section is useful for all stakeholders to quickly grasp the essence of the project.***

Functional Requirements: ***Detail the specific functionalities and features of the Easy Invoicing Application. This section will be of interest to the development team, project managers, and business stakeholders to ensure a shared understanding of the desired capabilities.***

Non-Functional Requirements: ***Outline the performance, security, usability, and other non-functional requirements of the application. This section is important for the development team, testers, and quality assurance professionals to ensure compliance with the specified criteria.***

Use Cases or User Stories: ***Describe specific scenarios or user interactions with the application. This section helps the development team and business stakeholders understand how the application will be used and what results are expected.***

Constraints and Assumptions: ***Identify any constraints, limitations, or assumptions that may impact the project. This section provides important context for the development team and project managers to plan and execute the project effectively.***

1.3 Product Scope

The scope of the billing system with integrated e-invoicing capabilities is to provide organizations with a centralized platform for managing all invoicing activities. The system will enable users to create and send invoices quickly and easily, reducing the likelihood of errors and improving overall efficiency. The e-invoicing integration will allow users to send invoices electronically, eliminating the need for physical copies and reducing the carbon footprint.

The system will include features such as automatic payment reminders, invoice tracking, and customizable invoice templates. The system will also provide users with insights into their billing and payment history, allowing for better financial management.

The scope of the system will be limited to the following:

Invoicing:

The system will provide users with a platform for creating, sending, and managing invoices.

Payment Processing:

The system will enable users to process payments and manage payment information.

Reporting and Analytics:

The system will provide users with insights into their billing and payment history.

Integration:

The system will be designed to be scalable and customizable, allowing for future upgrades and enhancements.

The system will be developed using the latest technology and programming languages to ensure maximum performance, security, and reliability.

Overall, the scope of the billing system with integrated e-invoicing capabilities is to provide organizations with a modern and efficient way to manage invoicing activities, improve financial management, and reduce the carbon footprint.
The system will be designed with the end-user in mind, making it user-friendly and customizable to meet the specific needs of different organizations.

2. Overall Description

2.1 Product Perspective

System Description: ***Provide an overview of the Easy Invoicing Application, including its purpose, main functions, and how it fits into the overall business ecosystem. Describe the application's role in the invoicing process and its relationship with other existing systems or components.***

System Boundaries: ***Clearly define the boundaries of the Easy Invoicing Application in terms of the components it includes and the interfaces it interacts with. Specify any external systems or services that the application needs to integrate with, such as payment gateways or customer databases.***

Interactions and Dependencies: ***Identify any dependencies or interactions between the Easy Invoicing Application and external systems, users, or processes. This could include data exchanges, communication protocols, or integration points with other applications.***

User Roles: ***Define the different user roles or personas who will interact with the Easy Invoicing Application. This could include administrators, invoice creators, clients, or accounting personnel. Describe their responsibilities, access levels, and specific functionalities they will have within the system.***

Hardware and Software Requirements: ***Specify the hardware infrastructure and software components required to support the Easy Invoicing Application. This could include server specifications, operating systems, web browsers, and any third-party software dependencies.***

System Interfaces: ***Identify the interfaces through which the Easy Invoicing Application will interact with external systems or users. This could include APIs, web services, user interfaces, or file formats used for data exchange.***

System Constraints: ***Outline any constraints or limitations that may impact the design or implementation of the Easy Invoicing Application. This could include technical constraints, regulatory requirements, or specific business rules that need to be considered.***

Product Functions

User Registration and Authentication: ***Provide functionality for users to register accounts and authenticate themselves to access the application securely.***

Invoice Creation: ***Allow users to create and generate invoices with customizable templates, including invoice details such as customer information, items or services provided, quantities, prices, and applicable taxes.***

Invoice Management: ***Enable users to manage their invoices, including viewing, editing, and deleting existing invoices. Provide search and filtering capabilities to locate specific invoices based on criteria such as date, customer name, or invoice number.***

Payment Processing: ***Facilitate payment processing by integrating with payment gateways or providing options for manual payment entry. This may include tracking payment status, generating payment reminders, and updating invoice statuses upon payment confirmation.***

Customer Management: ***Allow users to manage customer information, including creating and maintaining customer profiles, storing contact details, and tracking customer history.***

Reporting and Analytics: ***Provide reporting features that allow users to generate various reports, such as sales summaries, outstanding invoices, payment history, or customer analytics. Include visualization capabilities to present data in a clear and meaningful way.***

Tax Calculation: ***Support automatic tax calculation based on predefined tax rules and rates. This ensures accurate tax calculations on invoices and compliance with tax regulations.***

Integration with Accounting Systems: ***Enable integration with accounting systems or provide export capabilities for seamless transfer of invoicing data to accounting software.***

Notifications and Reminders: ***Implement notifications and reminders functionality to alert users about payment due dates, invoice status updates, or other relevant events.***

User Settings and Customization: ***Allow users to customize settings according to their preferences, such as language selection, date formats, currency, and invoice numbering schemes***

User Classes and Characteristics

1) ***Administrator:***

Characteristics: ***Has full access and control over the application.***

Responsibilities: ***Manages user accounts, sets permissions, configures system settings, and performs administrative tasks.***

2) ***Invoice Creator:***

Characteristics: ***Regular user responsible for creating and managing invoices.***

Responsibilities: ***Creates new invoices, edits existing invoices, adds line items, sets payment terms, and manages invoice lifecycle.***

3) ***Customer:***

Characteristics: ***External entity or client who receives invoices from the Easy Invoicing Application.***

Responsibilities: ***May have limited access to view and pay invoices, manage their own account details, and interact with the application as necessary.***

4) ***Accountant/Finance Personnel:***

Characteristics: ***User responsible for financial management and accounting tasks.***

Responsibilities: ***May have access to financial reports, payment records, and reconciliation functionality. They may also perform tasks related to integrating invoice data with accounting systems.***

5) ***System Administrator/IT Support:***

Characteristics: ***Responsible for managing the technical infrastructure and supporting the application.***

Responsibilities: ***Handles system maintenance, upgrades, backups, security configurations, and troubleshooting.***

2.2 Operating Environment

1) ***Hardware Requirements:***

Processor: ***Specify the minimum required processor type and speed.***

Memory (RAM): ***Specify the minimum required RAM for optimal performance.***

Storage: ***Specify the minimum required disk space for installation and data storage.***

2) ***Software Requirements:***

Operating System: ***Identify the specific operating systems supported by the application, such as Windows, macOS, or Linux, along with the required versions.***

Web Server: ***Specify the web server software, if applicable, that is compatible with the application.***

Database: ***Identify the database management system (DBMS) required for storing and retrieving data.***

Programming Languages: ***Specify the programming languages and versions necessary for running the application.***

Frameworks and Libraries: ***Identify any specific frameworks, libraries, or runtime environments required by the application.***

3) Network Requirements:

Internet Connectivity: ***Specify whether an internet connection is required for certain features or integrations.***

Ports and Protocols: ***Identify any specific network ports or protocols that need to be accessible for the application's functionality.***

4) Additional Dependencies:

Third-Party Software: ***Specify any third-party software or tools that are required for the application to function properly.***

APIs and Integrations: ***Identify any external APIs or integrations that the application relies on and specify any requirements for those integrations.***

2.3 Design and Implementation Constraints

Technology Constraints: ***Specify any specific technologies, frameworks, or programming languages that must be used for the development of the application. For example, if the application needs to be built using a specific programming language or a particular framework, mention those constraints.***

Compatibility Constraints*: Identify any compatibility requirements or constraints that the application must adhere to. This could include compatibility with specific web browsers, operating systems, or database management systems.*

Performance Constraints: ***Specify any performance requirements or limitations that the application needs to meet. For example, if the application must be able to handle a certain number of concurrent users or process a specific number of transactions within a given time frame, mention those constraints.***

Security Constraints: ***Identify any security requirements or constraints that the application must satisfy. This could include data encryption, secure authentication mechanisms, or compliance with specific security standards or regulations.***

Regulatory Constraints: ***Specify any regulatory requirements that the application must comply with. For example, if the application handles sensitive financial information, it may need to adhere to industry-specific regulations such as PCI-DSS (Payment Card Industry Data Security Standard).***

Integration Constraints: ***Identify any constraints related to integrating the Easy Invoicing Application with other systems or services. For example, if the application needs to integrate with external payment gateways or accounting software, mention those constraints.***

Time and Resource Constraints: ***Specify any constraints related to project timelines, budget, or available resources. This could include deadlines for specific milestones, limitations on development resources, or budget constraints.***

Usability Constraints: ***Identify any usability requirements or constraints for the application. This could include factors such as accessibility compliance, user interface guidelines, or specific user experience (UX) considerations.***

2.4 User Documentation

User Manuals/Guides: ***Create comprehensive user manuals or guides that provide step-by-step instructions on how to use the application. Include detailed explanations of each feature, including screenshots or illustrations to enhance understanding. Organize the information logically and make it easily accessible for users to find the relevant instructions.***

Installation Guide: ***Provide an installation guide that walks users through the process of installing and setting up the application on their systems. Include system requirements, software dependencies, and any configuration steps needed for successful installation.***

Getting Started Guide: ***Develop a getting started guide to introduce users to the application and its key features. Provide an overview of the user interface, basic navigation instructions, and essential functionalities to help users quickly become familiar with the application.***

FAQs and Troubleshooting: ***Compile a list of frequently asked questions (FAQs) and common troubleshooting scenarios. Address common user queries and provide solutions to potential issues they may encounter while using the application. This helps users find quick resolutions and reduces the need for support.***

Online Help or Contextual Help: ***Implement an online help system within the application that provides contextual assistance to users. This can include tooltips, pop-up help messages, or searchable help documentation accessible from within the application itself.***

Video Tutorials: ***Consider creating video tutorials that visually demonstrate how to perform specific tasks or use certain features of the Easy Invoicing Application. Videos can be effective in conveying complex processes and providing a more engaging learning experience.***

Release Notes: ***Include release notes for each version or update of the application. Highlight new features, bug fixes, and any changes that users need to be aware of.***

Glossary: ***Provide a glossary of terms and acronyms used in the application to assist users in understanding the terminology.***

Online Community or User Forums: ***Consider establishing an online community or user forums where users can interact, ask questions, and share their experiences with the Easy Invoicing Application. This can facilitate peer-to-peer support and knowledge sharing.***

Continuous Updates: ***Ensure that the user documentation stays up to date with the evolving features and functionality of the application. Regularly review and update the documentation to reflect any changes or enhancements.***

2.5 Assumptions and Dependencies

1) ***Assumptions:***

Assumed User Knowledge: ***Specify any assumptions about the level of knowledge or familiarity that users have with similar invoicing applications or basic accounting principles. For example, you may assume that users have a basic understanding of invoice generation and financial terminology.***

Availability of Internet Connection: ***Assume that users will have a stable internet connection to access and use the Easy Invoicing Application. This may be important for features such as online invoice delivery or real-time synchronization.***

User Hardware and Software Compatibility: ***Assume that users will have compatible hardware and software configurations that meet the minimum requirements specified in the operating environment section. This ensures that the application functions as intended on the user's system.***

2) ***Dependencies:***

Third-Party Integrations: ***Identify any external systems, APIs, or services that the Easy Invoicing Application depends on for specific functionalities. For example, if the application integrates with a payment gateway for online payments, specify the dependency on that payment gateway service.***

Database Management System: ***Specify the dependency on a specific database management system (DBMS) for storing and retrieving invoice data. This could be MySQL, PostgreSQL, or another DBMS.***

Development Frameworks and Libraries: ***Identify any dependencies on specific development frameworks, libraries, or modules that are required for the development and operation of the application. For example, if the application is built using a specific programming language or relies on a particular framework like Laravel or Django, mention those dependencies.***

Compliance with Legal and Regulatory Requirements: ***Specify any dependencies on complying with specific legal or regulatory frameworks. For example, if the application must adhere to data protection regulations like GDPR (General Data Protection Regulation) or tax regulations specific to a particular jurisdiction, highlight those dependencies.***

Availability of External Services: ***Identify any dependencies on external services, such as email delivery services or SMS gateways, for features like automated invoice notifications. This ensures that these services are available and accessible to the Easy Invoicing Application.***

3. External Interface Requirements

3.1 User Interfaces

Intuitive and User-Friendly Design: ***Create user interfaces that are intuitive and easy to navigate. Use clear and descriptive labels, icons, and visual cues to guide users through the application. Keep the interface clean and uncluttered to avoid overwhelming users with unnecessary information.***

Consistent Layout and Navigation: ***Maintain consistency in the layout and navigation across different screens of the application. Use a standardized menu or navigation bar to provide easy access to different sections or functionalities. Consistency helps users quickly learn and understand the application's structure.***

Responsive Design: ***Ensure that the user interfaces are responsive and adapt well to different screen sizes and devices. The application should be accessible and usable on desktops, laptops, tablets, and mobile devices. Consider responsive design principles to optimize the user experience across various platforms.***

Clear Hierarchy and Organization: ***Organize information and features in a logical and hierarchical manner. Group related elements together, and use visual cues such as headings, sections, and tabs to differentiate between different areas of functionality. This helps users find what they need efficiently and reduces cognitive load.***

Input Validation and Error Handling: ***Implement input validation to provide immediate feedback to users when they enter incorrect or incomplete information. Clearly indicate any validation errors and provide helpful error messages to assist users in correcting their input. Consider using tooltips or inline validation to guide users while they fill out forms.***

Contextual Help and Tooltips: ***Provide contextual help and tooltips to assist users in understanding specific features or actions. These can be triggered when users hover over an element or click on a help icon. Contextual help improves user comprehension and reduces the need to search for information.***

Clear Call-to-Action Buttons: ***Use clear and descriptive labels for buttons and calls-to-action. Make them visually prominent and easily distinguishable. Proper placement and design of buttons help users understand their purpose and encourages them to take the desired actions.***

Visual Feedback: ***Provide visual feedback to users when they perform actions or interact with the application. This can include highlighting selected items, displaying progress indicators, or showing success/error messages. Visual feedback enhances the user experience and helps users understand the application's response to their actions.***

Customization Options: ***Consider providing customization options, where appropriate, to allow users to personalize their experience. For example, users may have the ability to choose their preferred color scheme or customize the layout of their dashboard.***

Accessibility Considerations: ***Ensure that the user interfaces meet accessibility standards and guidelines, making the application usable by a wide range of users. Consider factors such as color contrast, keyboard navigation, screen reader compatibility, and support for assistive technologies.***

3.2 Hardware Interfaces

Printers: ***The application may need to interact with printers to generate physical copies of invoices or reports. It should support standard printer interfaces, such as USB or network connectivity, to enable users to print documents directly from the application.***

Scanners: ***If the application allows users to scan and attach physical documents to invoices or records, it should support the integration of scanners. This enables users to scan documents and import them into the application for further processing or storage.***

Barcode Scanners: ***If the application supports barcode scanning for inventory management or product identification, it should be compatible with barcode scanners. These scanners can be connected via USB or Bluetooth and allow users to scan barcodes to quickly input data into the application.***

Card Readers: ***If the application incorporates payment processing and accepts credit or debit card payments, it may need to interface with card readers. The card readers can be connected via USB or Bluetooth and facilitate secure card transactions within the application.***

Mobile Devices: ***If the Easy Invoicing Application has a mobile version or supports mobile access, it should be compatible with various mobile devices such as smartphones or tablets. The application should adapt to different screen sizes and utilize the device's hardware capabilities, such as cameras for scanning or GPS for location-based services.***

Barcode Printers: ***If the application generates and prints barcode labels for products or inventory, it should support integration with barcode printers. These printers are used to print labels containing barcodes that can be scanned for inventory management purposes.***

Cash Registers: ***If the Easy Invoicing Application is used in retail or point-of-sale environments, it may need to integrate with cash registers or cash drawers. This allows for the synchronization of sales data, opening and closing cash registers, and managing cash transactions.***

3.3 Software Interfaces

Payment Gateway: ***If the application supports online payment processing, it may need to integrate with a payment gateway service. The payment gateway acts as an intermediary between the application and payment processors, facilitating secure and reliable online payment transactions. Examples of payment gateway providers include PayPal, Stripe, or Authorize.Net. The application should have the necessary integration capabilities and APIs to communicate with the chosen payment gateway.***

Email Service Provider: ***The Easy Invoicing Application may require integration with an email service provider to send automated email notifications, such as invoice receipts, payment reminders, or account statements. Popular email service providers include Gmail, SendGrid, or Mailchimp. The application should support the required protocols (e.g., SMTP) and APIs to connect and send emails through the chosen provider.***

SMS Gateway: ***If the application supports SMS notifications or alerts, it may need to integrate with an SMS gateway service. This allows the application to send SMS messages to customers or users for important updates or reminders. Examples of SMS gateway providers include Twilio, Nexmo, or Plivo. The application should have the necessary integration capabilities and APIs to communicate with the chosen SMS gateway.***

File Storage and Cloud Services: ***The Easy Invoicing Application may need to integrate with file storage or cloud services to store and retrieve documents, such as scanned receipts, invoice attachments, or backup files. Commonly used services include Amazon S3, Google Drive, or Microsoft OneDrive. The application should support the required protocols (e.g., SFTP, REST APIs) to interact with the chosen file storage or cloud service.***

Third-Party APIs: ***Depending on the specific requirements and functionalities of the Easy Invoicing Application, it may need to integrate with various third-party APIs, such as shipping services (e.g., UPS, FedEx), tax calculation services (e.g., Avalara, TaxJar), or customer relationship management (CRM) systems (e.g., Salesforce, HubSpot). The application should have the necessary integration capabilities and API documentation for seamless communication with these external services.***

3.4 Communications Interfaces

HTTP(S) and RESTful APIs: ***The application may communicate with external systems or services using the Hypertext Transfer Protocol (HTTP) or its secure variant HTTPS. RESTful APIs can be utilized to exchange data and perform operations with remote resources over the internet. The application should define and adhere to the required API endpoints, HTTP methods (e.g., GET, POST, PUT, DELETE), and data formats (e.g., JSON, XML) for communication.***

Webhooks: ***Webhooks allow the Easy Invoicing Application to receive real-time notifications or events from external systems. These notifications can trigger actions or updates within the application. The application should define the necessary webhook endpoints and specify the expected payload format and authentication mechanisms to receive and process incoming webhook data.***

SMTP and Email Protocols: ***If the application involves sending or receiving emails, it needs to interact with Simple Mail Transfer Protocol (SMTP) servers or other email protocols. The application should support the required configuration parameters (e.g., SMTP server address, port, authentication) to enable email communication.***

Messaging Protocols: ***The Easy Invoicing Application may need to integrate with messaging protocols or services for real-time communication or notifications. Examples include the Message Queuing Telemetry Transport (MQTT) protocol for Internet of Things (IoT) applications or the Simple Notification Service (SNS) provided by cloud service providers. The application should support the necessary configuration and connection parameters for seamless messaging integration.***

FTP/SFTP and File Transfer Protocols: ***If the application involves file transfer operations, it may need to support File Transfer Protocol (FTP) or its secure variant, SSH File Transfer Protocol (SFTP). These protocols enable the application to transfer files to and from remote servers or storage systems. The application should handle the necessary authentication, encryption, and file transfer operations according to the chosen protocol.***

Web Services: ***The Easy Invoicing Application may need to integrate with web services, such as SOAP-based services or XML-RPC services. These services allow the application to exchange structured data and perform remote procedure calls. The application should support the required protocols and data formats specified by the web services it interacts with.***

4. System Features

4.1 System Feature 1

System User: Admin – Customer- Seller.

Admin:

- ***As an admin, I want to edit System user types***
- ***As an admin, I want to edit System products***
- ***As an admin, I want to edit Products price.***

Seller:

- ***As a seller, I want to have an account***
- ***As a seller, I want to Login***
- ***As a seller, I want to add my location***
- ***As a seller, I want to determine my products***
- ***As a seller, I want to update the products price***
- ***As a seller, I want to give invoice to the user***
- ***As a seller, I want to receive the products status after selling***
- ***as a seller, I want to receive a calculation on purchased products & my profit***
- ***As a seller, I want to view the added review on my products.***

Customer:

- ***As a Customer, I want to have an account (Optional)***
- ***As a Customer, I want to Login (Optional)***
- ***As a Customer, I want to add my location***
- ***As a Customer, I want to View available products today in my location with their price***
- ***As a Customer, I want to select from the available products***

- ***As a Customer, I want to order the selected products***
- ***As a Customer, I want to receive an invoice of purchased products***
- ***As a Customer, I want to receive my order in the identified location***
- ***As a Customer, I want to add review on my order.***

5. Other Nonfunctional Requirements

Response Time: ***Specify the maximum acceptable response time for different operations or actions within the application. For instance, the system should provide a response within 2 seconds when generating an invoice or retrieving customer information***

Throughput: ***Define the desired number of transactions or operations that the system should be able to handle within a given time period. This requirement may specify the number of invoices the system should be able to process per minute or the number of concurrent users it can support.***

Scalability: ***Outline the expected scalability requirements, such as the ability to handle an increasing number of users or a growing volume of data without significant performance degradation. This may include specifying the expected growth rate and the maximum number of concurrent users the system should support.***

Reliability: ***Define the desired system uptime and availability. This requirement may specify the maximum allowable downtime or the expected uptime percentage over a certain period. For example, the system should be available 99% of the time during business hours.***

Resource Usage: ***Specify the expected resource consumption, such as CPU, memory, and network bandwidth. This requirement ensures that the application operates efficiently and does not excessively consume system resources.***

5.1 References

1) "Wave Invoicing" - Wave offers a free and user-friendly online invoicing platform that allows you to create, customize, and send professional invoices. You can find more information on their website: https://www.waveapps.com/invoicing/

2) "QuickBooks Invoicing" - QuickBooks provides a popular invoicing solution that is widely used by businesses of all sizes. It offers features such as invoice creation, tracking, and payment management. You can learn more on their website: https://quickbooks.intuit.com/invoicing/

3) "FreshBooks" - FreshBooks is another invoicing software that simplifies the invoicing process with easy-to-use templates, automated reminders, and online payment options. You can find more information here: https://www.freshbooks.com/invoicing

4) "Zoho Invoice" - Zoho Invoice offers an intuitive invoicing software with features like customizable templates, automated workflows, and online payment integration. You can learn more about their invoicing solution here: https://www.zoho.com/invoice/