A
Synopsis Report
on

Vendor Invoice tracker for Small and Medium scale companies

TY COMP
in
Computer Engineering
by

Bakliwal Aagam 112003013
Anushka Vijaykumar Naik 112003010
Ninad Barve 112003016

Department of Computer Engineering

Department of Computer Engineering,
COEP Technological University (COEP Tech)
(A Unitary Public University of Govt. of Maharashtra)
Shivajinagar, Pune-411005,Maharashtra, INDIA

Project Title : Vendor Invoice tracker for Small and Medium scale companies

Problem Statement

As a part of any transaction between an vendor and a client, the vendor has to generate an invoice and send it to the client. In small companies, this process is usually managed by an Accounting Executive who receives the Invoice in either hard copy or via mail. Vendor invoice tracking is the process of monitoring and managing the flow of invoices from vendors to an organization. The goal is to ensure that all invoices are received, approved, and processed in a timely and efficient manner, with minimal errors or discrepancies. This process is critical for maintaining accurate financial records and tracking vendor payments, and can be a significant challenge for organizations that deal with a high volume of invoices

Hardware and Software Requirements

Hardware Requirements

-Processor requirement : A processor with a minimum of 4 cores and a clock speed of 2.0 GHz or higher
-Memory requirement: Minimum of 4GB RAM
-For best results: 8GB RAM
-Connection requirement: A reliable internet connection with a high-speed bandwidth

Software Requirements

1. Python: You will need to have Python installed on your system to run Flask, which is a Python-based web framework.
2. Flask: This is a lightweight web framework for Python that allows you to build web applications quickly and easily.
3. SQLite: SQLite is a lightweight and portable relational database management system that is suitable for small to medium-sized web applications.
4. JavaScript: JavaScript is a programming language that is commonly used for front-end web development. It allows you to add interactivity and dynamic content to web pages.
5. HTML: HTML is a markup language that is used to structure and display content on the web. It provides the basic structure of web pages.
6. CSS: CSS is a style sheet language that is used to control the presentation and layout of web pages.

7. Web server: You will need a web server to host and serve your web application. You can use a variety of web servers, including Apache, Nginx, or even the built-in development server that comes with Flask.

8. Operating system: You will need an operating system to run your web server and database. Flask, SQLite, JavaScript, HTML, and CSS are all cross-platform, so you can use them on Windows, macOS, or Linux.

9. Text editor or integrated development environment (IDE): You will need a text editor or IDE to write your code. Some popular options include Visual Studio Code, PyCharm, and Sublime Text.

Architectural Diagram

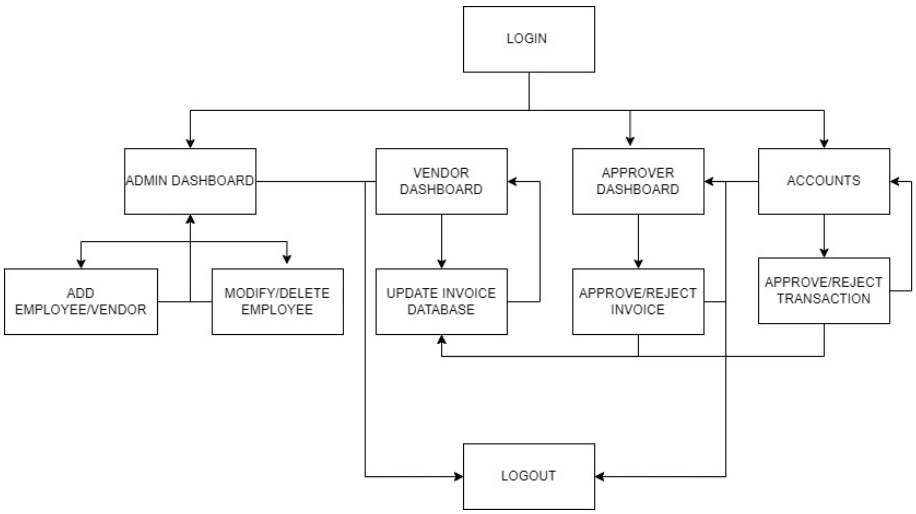


Figure 1: Block diagram of the system

Process Model

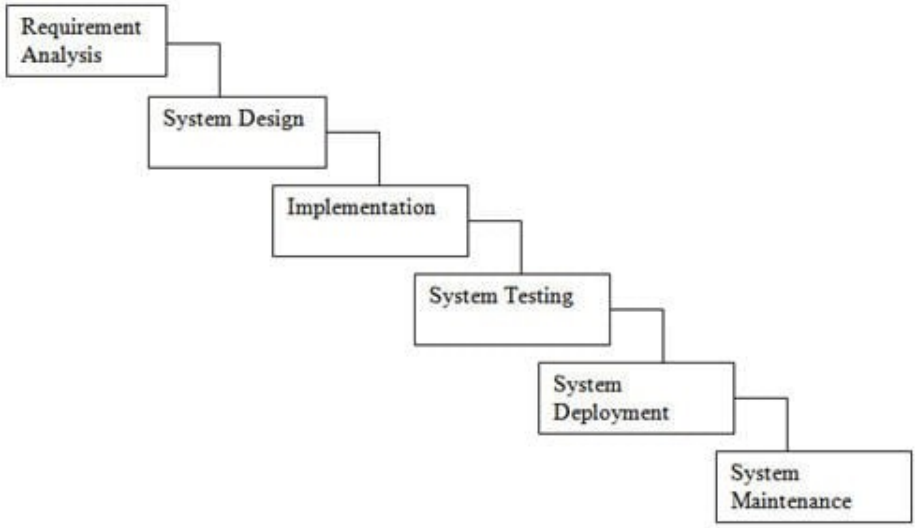
Requirement gathering: the first step will be to require the vendors, clients and accounts department to make accounts and login to the portal and add their payments and transactions fulfiled in order to track and update status of the invoices.
Design: the software design phase will contain designing the architecture and pipeline connecting the different users and the actions different entities will perform as well as design the user interface and database.
Development: The development phase will involve writing the code for the system, including the database, website and backend.

Deployment: The deployment phase will involve deploying the system to the internet and making it accessible to everyone with a valid account and internet connection.
Maintenance: The system will require ongoing maintenance to ensure that it continues to meet the needs of vendors, and continue smooth integration of the accounts department as well as vendors and clients and updates if needed.

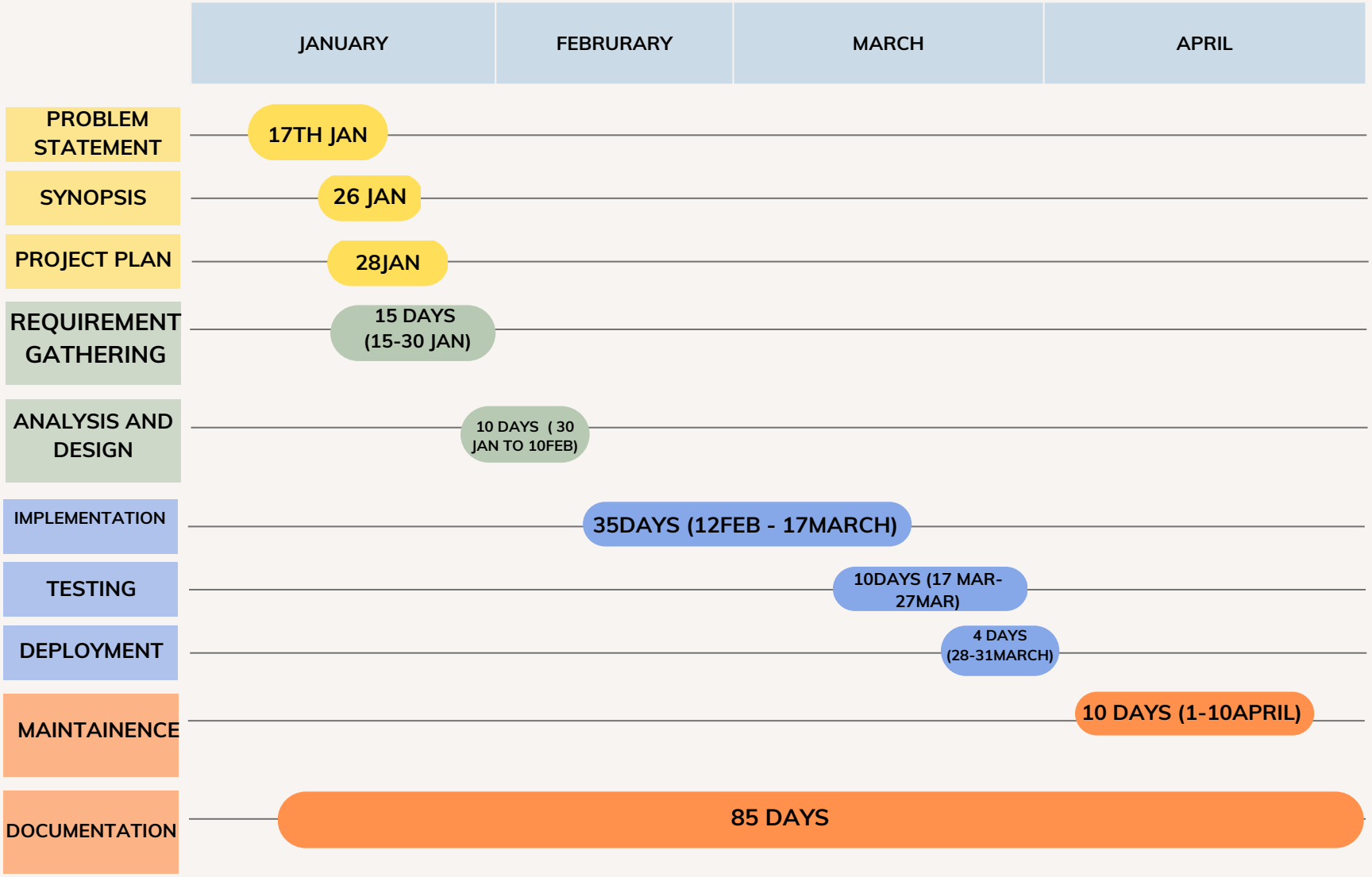
The waterfall model is a sequential, plan driven-process where you must plan and schedule all your activities before starting the project. Each activity in the waterfall model is represented as a separate phase arranged in linear order.

It has the following phases:

Requirements, Design, Implementation, Testing, Deployment, Maintenance,



GANTT CHART : SOFTWARE ENGINEERING MINI PROJECT-II



SOFTWARE REQUIREMENTS SPECIFICATION

for

Vendor Invoice tracker for Small and Medium scale companies

Version 1.0

Prepared by : 1. Anushka Vijaykumar Naik (112003010)
2. Ninad Barve (112003016)
3. Bakliwal Aagam (112003013)

Submitted to : Tanuja Pattanshetty
Professor

Contents

1 Introduction

1.1 Purpose

As a part of any transaction between an vendor and a client, the vendor generally invoices the client, the client then forwards the invoice to the concerned authority for approving the payment for the invoice, which is then sent to the accounts department. Following this, the money is disbursed to the vendor and a confirmation receipt/email is issued to him. The issue in the whole process is that it is a black-box, i.e. the vendor is unaware of the status of the invoice with the organization and hence to to repetitively follow up with the client to check the status of the invoice, which burdens the client with redundant emails/communication and also is not the best utilization of time and energy for the vendor. To ensure transparency in the process, the "Vendor Invoice Tracker for Small and Medium Scale Companies" is being created.

1.2 Intended Audience and Reading Suggestions

This SRS is for developers, project managers, users and testers. Further the discussion will provide all the internal, external, functional and also non-functional information about the "Vendor Invoice tracker for Small and Medium scale companies".

1.3 Project Scope

"Vendor Invoice tracker for Small and Medium scale companies" creates a framework to allow transparency and create automation in the approval and disbursement of money to vendors. The project scope of your vendor invoice system includes the automation of the invoice processing workflow, real-time progress tracking, which in the long term will expedite the transaction system and make it more efficient as well as be beneficial for the vendors as they can track progress of each transaction in real time.

2 Overall Descriptions

2.1 Product perspective

The Vendor Invoice tracker for Small and Medium scale companies is a standalone software capable of handling and automating the entire vendor invoice generation system which eases the job of the company and adds real time tracking for vendors.

2.2 Product functions

The major functions of our project are as follows:

- Invoice creation: The system should allow the vendor to create an invoice and upload it to the system.
- Approval workflow: The system should have an approval workflow that allows the approver to review and approve the invoice before it is sent to the accounts department.
- Transaction processing: The system should enable the accounts department to complete the transaction and pay the vendor.
- Real-time progress tracking: The system should allow the vendor to track the progress of their invoice in real-time, from creation to payment.

2.3 User classes and Characteristics

- Admin: An employee who will create the initial accounts and add all the vendors and employees as well as be able to modify and delete the employee and their details
- Vendor: For creating, uploading and tracking the invoice in the system
- Approver: An employee of a company who will scrutinise each invoice and approve or reject them
- Accounts: An employee of a company who will approve or reject transactions and update the system accordingly

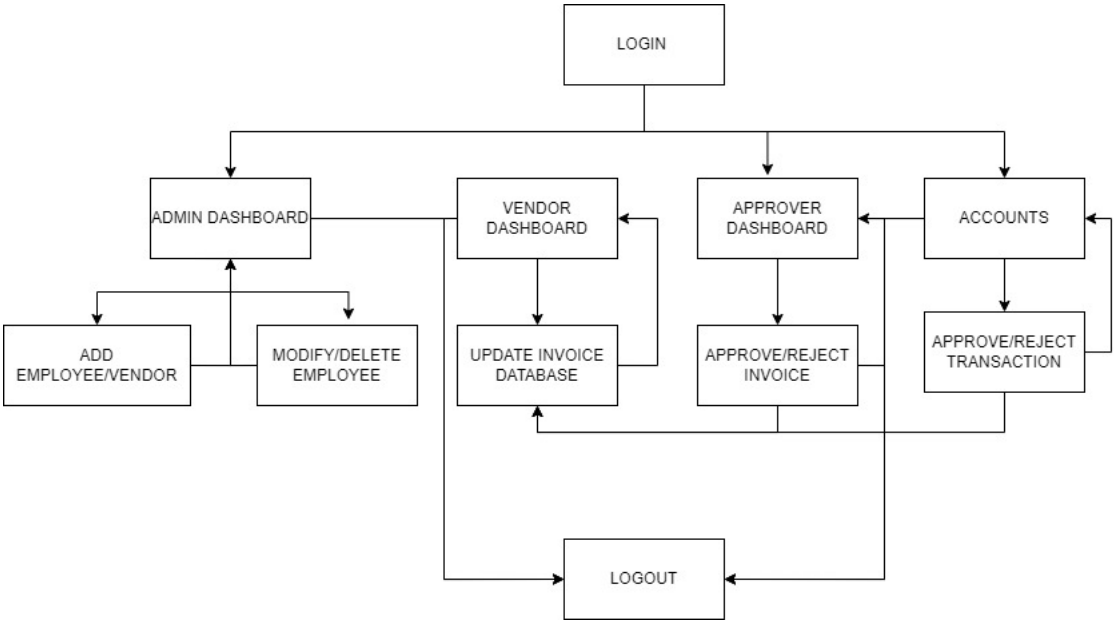


Figure 2.1: architectural diagram

2.4 Design and Implementation Constraints

1. Security: The system must be designed to ensure the security of sensitive information, such as client and vendor details, invoices, and payment information. The system should use secure authentication mechanisms, access control, and encryption to prevent unauthorized access and data breaches.

2. Scalability: The system should be designed to handle large volumes of invoices and users. It should be able to scale up or down as needed to meet changing demand.

3. Compatibility: The system should be designed to work with existing software and hardware systems used by the client, such as accounting software, payment gateways, and databases.

4. Usability: The system should be user-friendly and easy to navigate for both vendors and clients. It should have clear instructions and intuitive interfaces to minimize errors and improve user adoption.

5. Reliability: The system should be designed to minimize downtime and ensure consistent performance. It should have backup and disaster recovery mechanisms in place to prevent data loss and minimize disruptions.

3 External Interface Requirements

3.1 User Interfaces

There are functionalities such as the login page common for all users. There is an admin dashboard which allows the admin to add employees and vendors, modify employee details,delete employees and view all vendors and employees. There is a dedicated accounts dashboard which allows the accounts department to accept/reject transactions. There is an approver dashboard which allows the approver to approve or reject the invoices as well

3.2 Hardware Interfaces

- RAM: 8GB or more would be optimal
- Storage : Database requires storage of 100GB for a medium sized establishment
- Processor : i5 or above
- Internet : High speed stable internet for data consistency

3.3 Software Interfaces

1. Database : SQLite database interface
2. Frontend interface: Javascript,HTML,CSS using Flask micro service
3. Web server interface:Since we use Flask,we will use the WSGI interface to communicate with the web server
4. Flask-Mail extension: to interface with the SMTP server for sending email notifications.

4 System Features

4.1 Processing Narrative

4.1.1 Process Flow

Vendor Portal

1. Vendors log in to the portal using their credentials.
2. Vendors submit invoices by uploading them in the portal.
3. The system validates the invoice and sends a notification to the vendor if there are any errors or issues.
4. The system stores the invoice in the database and assigns a unique identifier to it.
5. The vendor can view the status of their submitted invoices and communicate with the client through the portal.

Client Portal

1. Clients log in to the portal using their credentials.
2. The system displays a dashboard showing the status of all invoices, including the ones that are pending, in review, approved, or rejected.
3. The system sends automated notifications to the client when an invoice is submitted, approved, or rejected.
4. The client can view details of each invoice, including the vendor name, invoice amount, due date, and status.
5. The client can communicate with vendors through the portal to resolve any issues or ask for more information.

Invoice Processing

1. The system checks the validity of the invoice, including the invoice number, vendor name, invoice date, and amount.
2. If the invoice is valid, the system assigns it to an approver based on the predefined approval workflow.

3. The approver reviews the invoice and either approves or rejects it based on predefined criteria, such as budget, policy compliance, or purchase order matching.
4. If the invoice is approved, the system generates a payment request and sends it to the finance department.
5. The finance department processes the payment and updates the invoice status in the system.

System Administration

1. The system administrator logs in to the administration portal using their credentials.
2. The system administrator can manage user accounts, including creating, editing, or deleting user profiles.
3. The system administrator can configure system settings, such as notification preferences, approval workflows, or data retention policies.
4. The system administrator can monitor system performance, including user activity, system logs, and system health.

4.1.2 Performance Characteristics

1. Response Time: The system should respond quickly to user requests, such as searching for invoices or generating reports. The response time should be within an acceptable range, typically measured in seconds or milliseconds.
2. Throughput: The system should be able to process a high volume of invoices and transactions per unit of time, without significant degradation in performance. Throughput is typically measured in transactions per second or requests per minute.
3. Concurrency: The system should be able to handle multiple users or transactions simultaneously, without causing delays or conflicts. The level of concurrency depends on the expected usage patterns and user behavior.
4. Scalability: The system should be able to scale up or down as needed to handle changes in demand, such as seasonal fluctuations or growth in the user base. Scalability can be achieved through various techniques, such as load balancing, clustering, or cloud computing.
5. Availability: The system should be available to users whenever they need it, without significant downtime or disruptions. The level of availability depends on the business requirements and service level agreements.
6. Reliability: The system should be able to perform its functions correctly and consistently, without errors or failures. Reliability can be measured by metrics such as mean time between failures (MTBF) or mean time to repair (MTTR).

5 Other Nonfunctional Requirements

5.1 Performance Requirements

1. Response Time: The system should respond to user requests within 3 seconds. The response time should be consistent even during peak usage hours.
2. Concurrent Users: The system should support a minimum of 50 concurrent users. The system should be scalable to support up to 200 concurrent users.
3. Data Storage: The system should be able to store at least 5 years of invoice data. The system should be scalable to support up to 10 years of invoice data.

5.2 Safety Requirements

1. Data security:
   The system must ensure the confidentiality, integrity, and availability of all data, including vendor and payment information. You should implement security measures such as access control, encryption, and regular backups to protect against data breaches and loss.
2. Data Backup and Recovery:
   The system should have a robust data backup and recovery mechanism in place to ensure that data is not lost in case of a system failure. The backup should be done daily, and the recovery time should not exceed four hours.

5.3 Security Requirements

Security is a critical aspect of any software application, and the Invoice Tracking System is no exception. The system deals with sensitive financial information, and any compromise in security can lead to serious consequences. In this section, we will discuss the security requirements of an invoice tracking system for small and medium-scale companies.

1. Authentication and Authorization:
   The system should have a robust authentication mechanism that ensures that only authorized users can access the system. The authentication mechanism should support strong passwords and two-factor authentication to prevent unauthorized access. The system should also have a role-based authorization mechanism that controls access to different parts of the system based on the user's role.

2. Data Encryption:
   The system should encrypt all sensitive data, including user passwords, invoice data, and customer information. The encryption should be done using industry-standard encryption algorithms and should be implemented in a way that ensures the confidentiality and integrity of the data.

3. Access Control:
   The system should have a well-defined access control mechanism that restricts access to sensitive data to authorized users only. The system should allow administrators to control access to different parts of the system based on the user's role. The system should also log all user activities to enable auditing and accountability.

4. System Security:
   The system should be designed with security in mind, and all security measures should be implemented at every level of the system. The system should be protected from common attacks such as SQL injection, cross-site scripting, and cross-site request forgery. The system should also be regularly tested for vulnerabilities, and all vulnerabilities should be addressed promptly.

5. Security Auditing:
   The system should have a security auditing mechanism that records all user activities, including login attempts, data access, and data modifications. The security auditing mechanism should also monitor the system for any suspicious activities, and alert administrators if any suspicious activities are detected.

6. Compliance:
   The system should comply with all relevant laws and regulations related to financial data management, such as the General Data Protection Regulation (GDPR) and the Payment Card Industry Data Security Standard (PCI-DSS).

5.4 Software Quality Attributes

1. Usability:
   Usability is the ease of use of a software application. The Invoice Tracking System should have an intuitive and user-friendly interface that is easy to navigate. The system should have clear and concise instructions and should provide feedback to users on their actions. The system should also have an efficient workflow that reduces the time required to complete tasks.

2. Reliability:
   Reliability is the ability of a software application to perform its intended function without failure. The Invoice Tracking System should be reliable and should be available to users at all times. The system should be able to handle errors gracefully and should have a mechanism in place to recover from failures. The system should also have a backup and recovery mechanism to ensure that data is not lost in case of a system failure.

3. Maintainability:
   Maintainability is the ease with which a software application can be maintained and updated. The Invoice Tracking System should be easy to maintain and update. The system should have clear and concise documentation that allows developers to understand the code and make changes efficiently. The system should also have a version control mechanism that allows developers to track changes to the code.

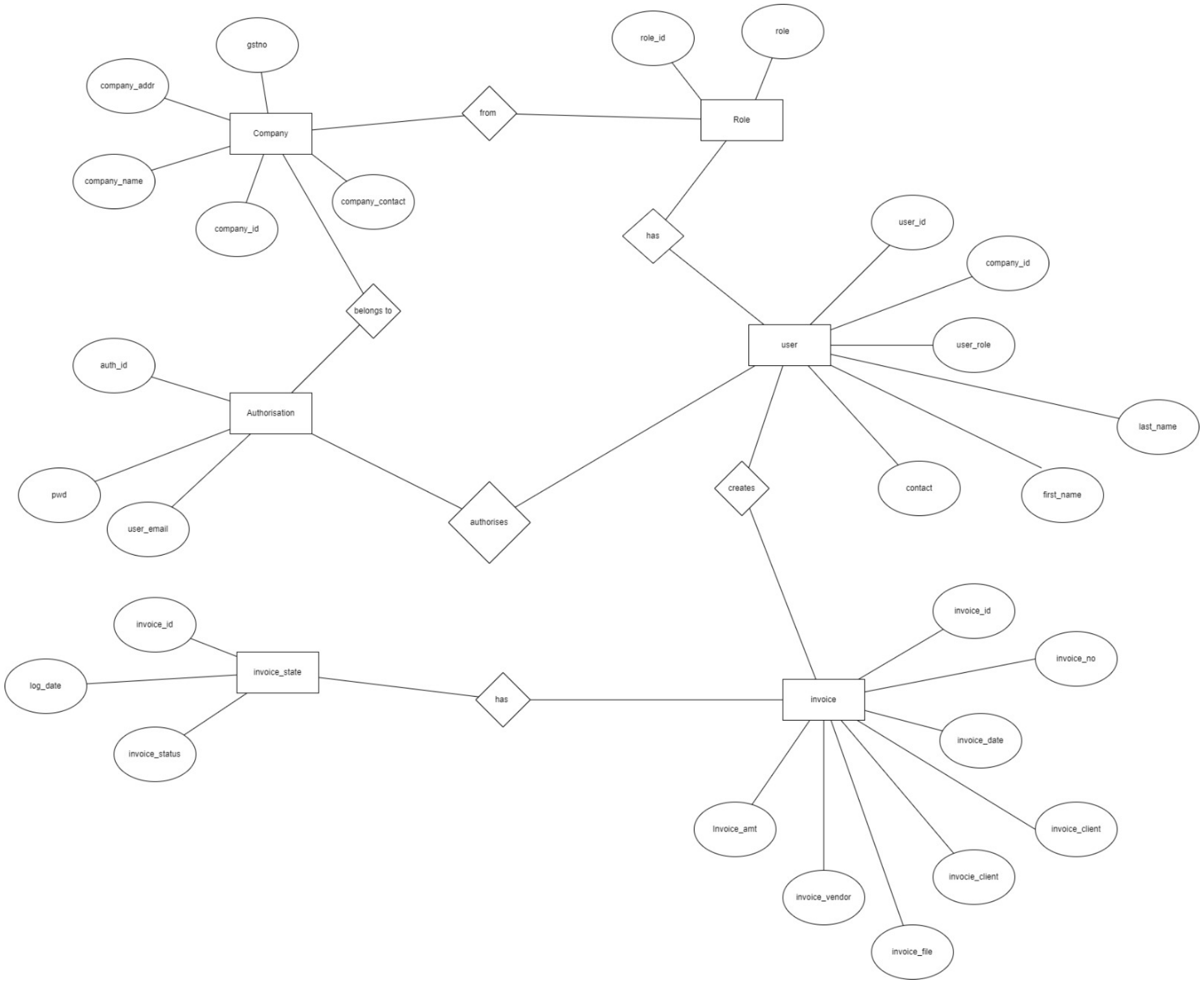
4. Testability:
   Testability is the ease with which a software application can be tested. The Invoice Tracking System should be testable and should have a comprehensive test suite that covers all aspects of the system. The system should also have a mechanism in place to automate testing to reduce the time required to test the system.

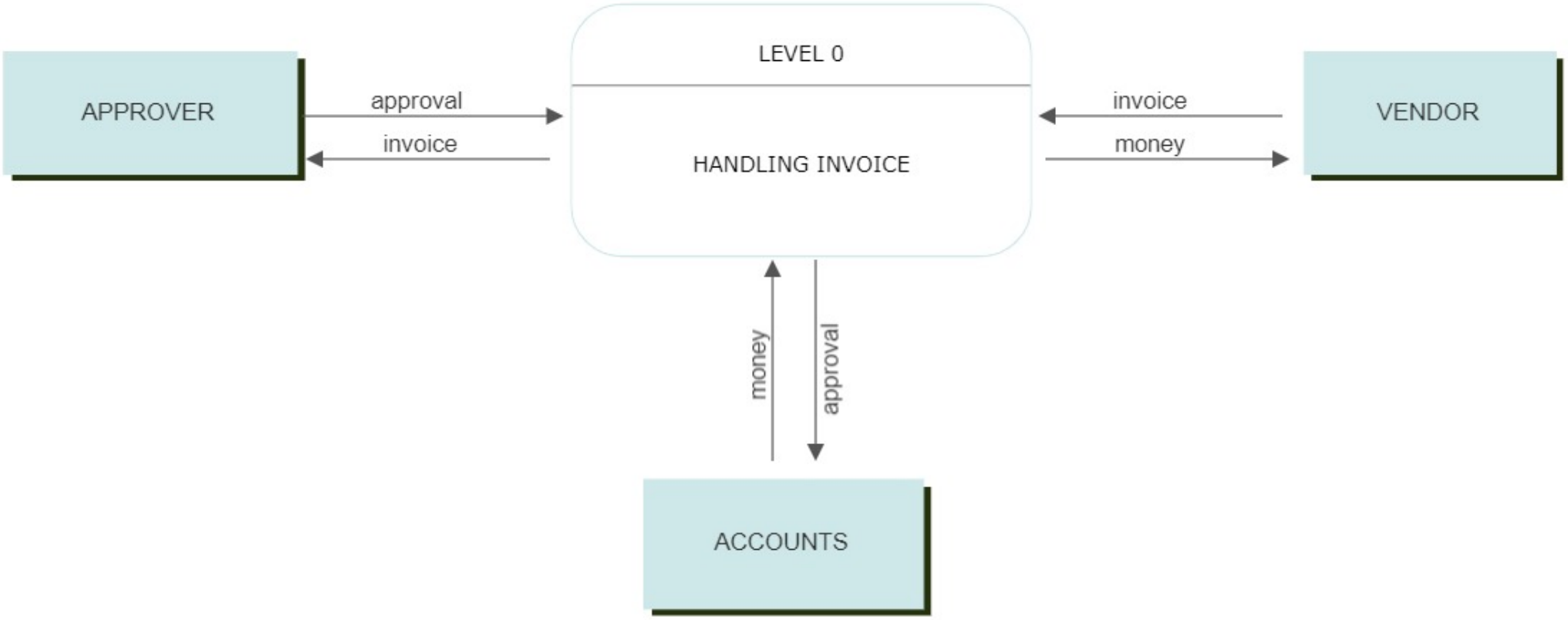
6 Other requirements

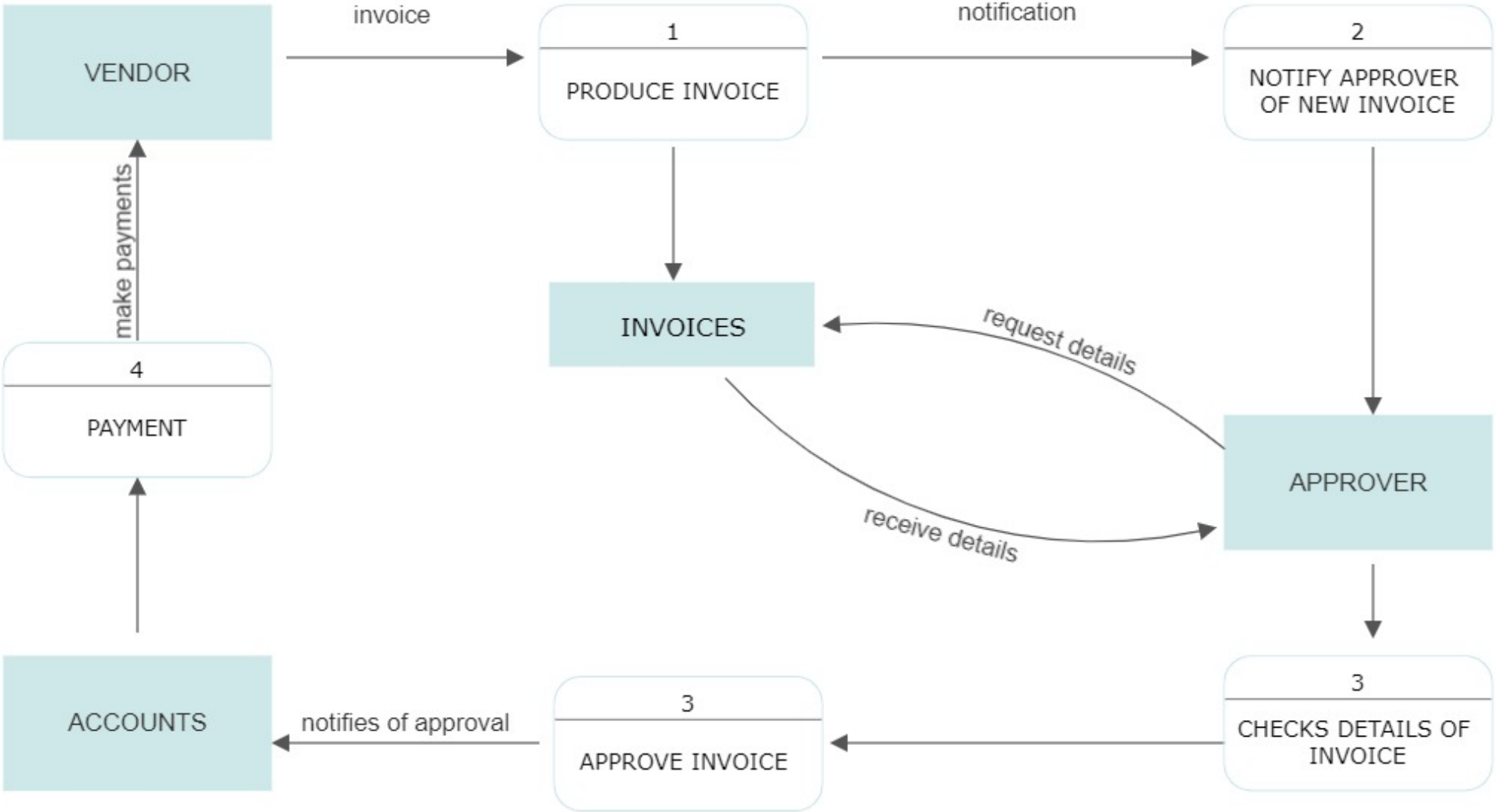
- Customization: Your system may need to be customized to meet the specific needs of your organization or industry. This could include adding custom fields, workflows, or integrations.
- Maintenance and support: Your system will require ongoing maintenance and support to ensure that it continues to operate effectively. This may include regular updates, bug fixes, and user support.

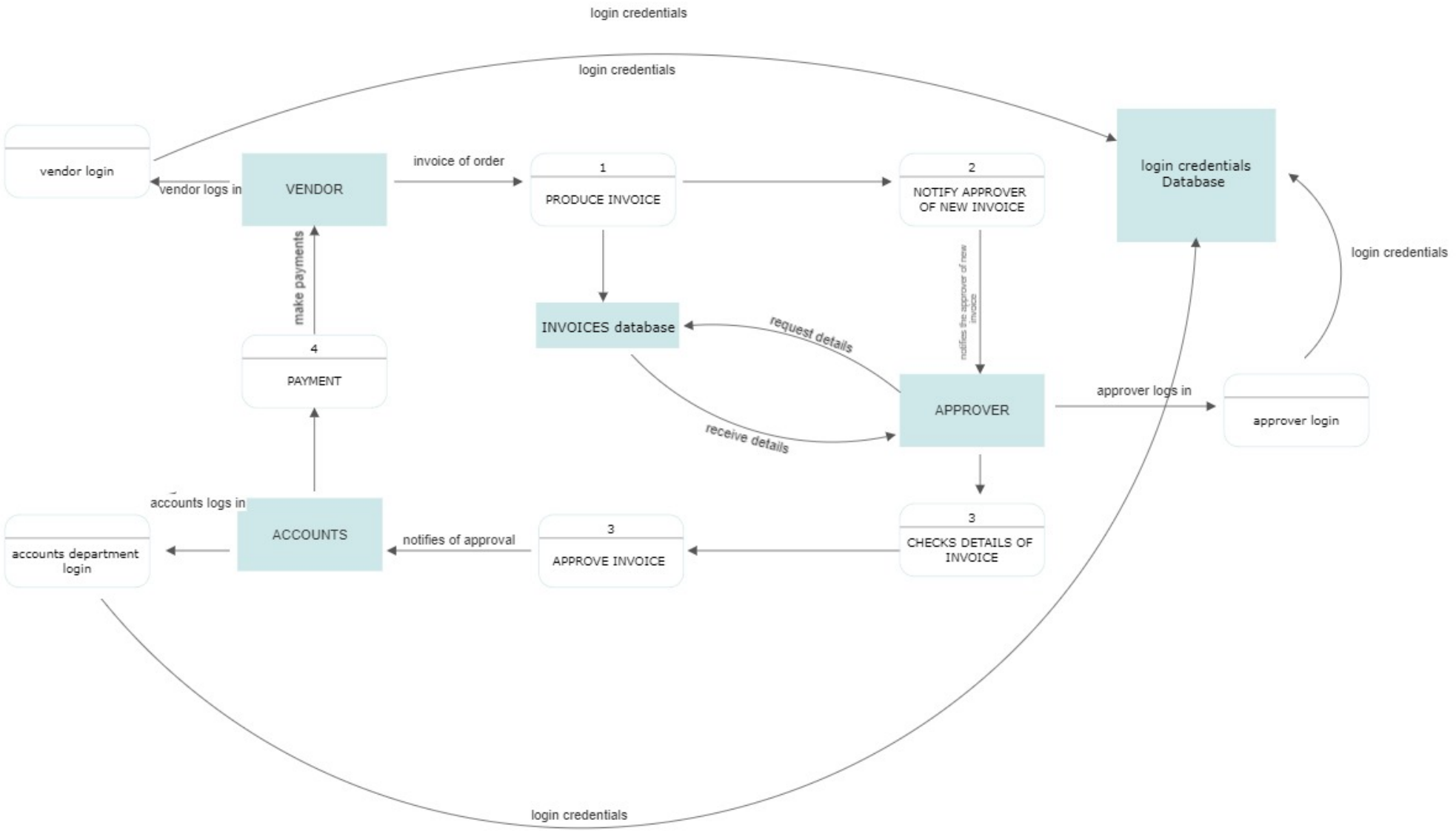
7 Appendix A: Glossary

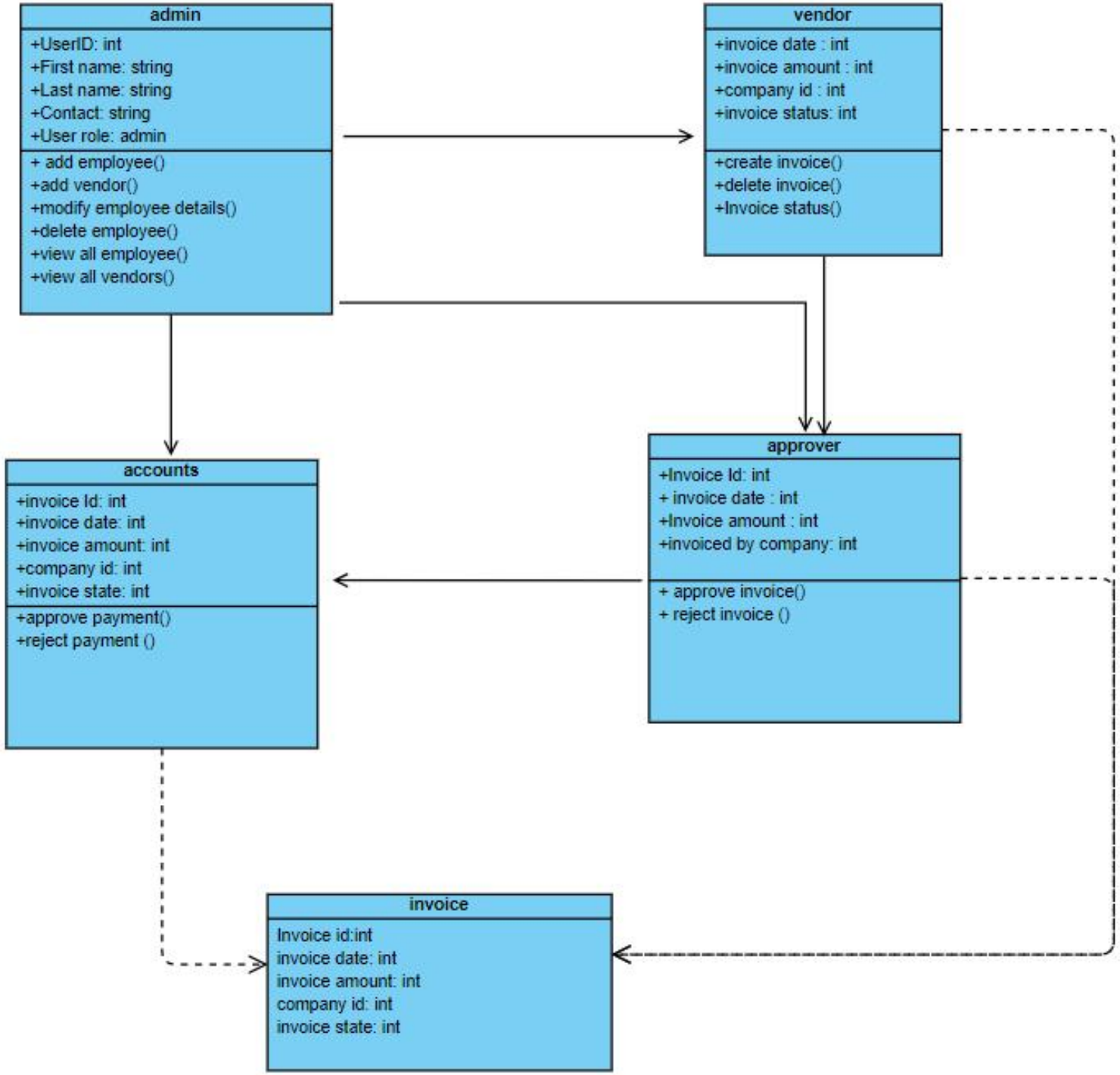
- Throughput: The quantity of information or materials that is put through a process in a specific period of time.
- SQLite: SQLite is a C-language library that implements a small, fast, self-contained, high-reliability, full-featured, SQL database engine
- ERD: An entity relationship diagram is a graphical representation that depicts relationships among people, objects, places, concepts or events within an information technology (IT) system.

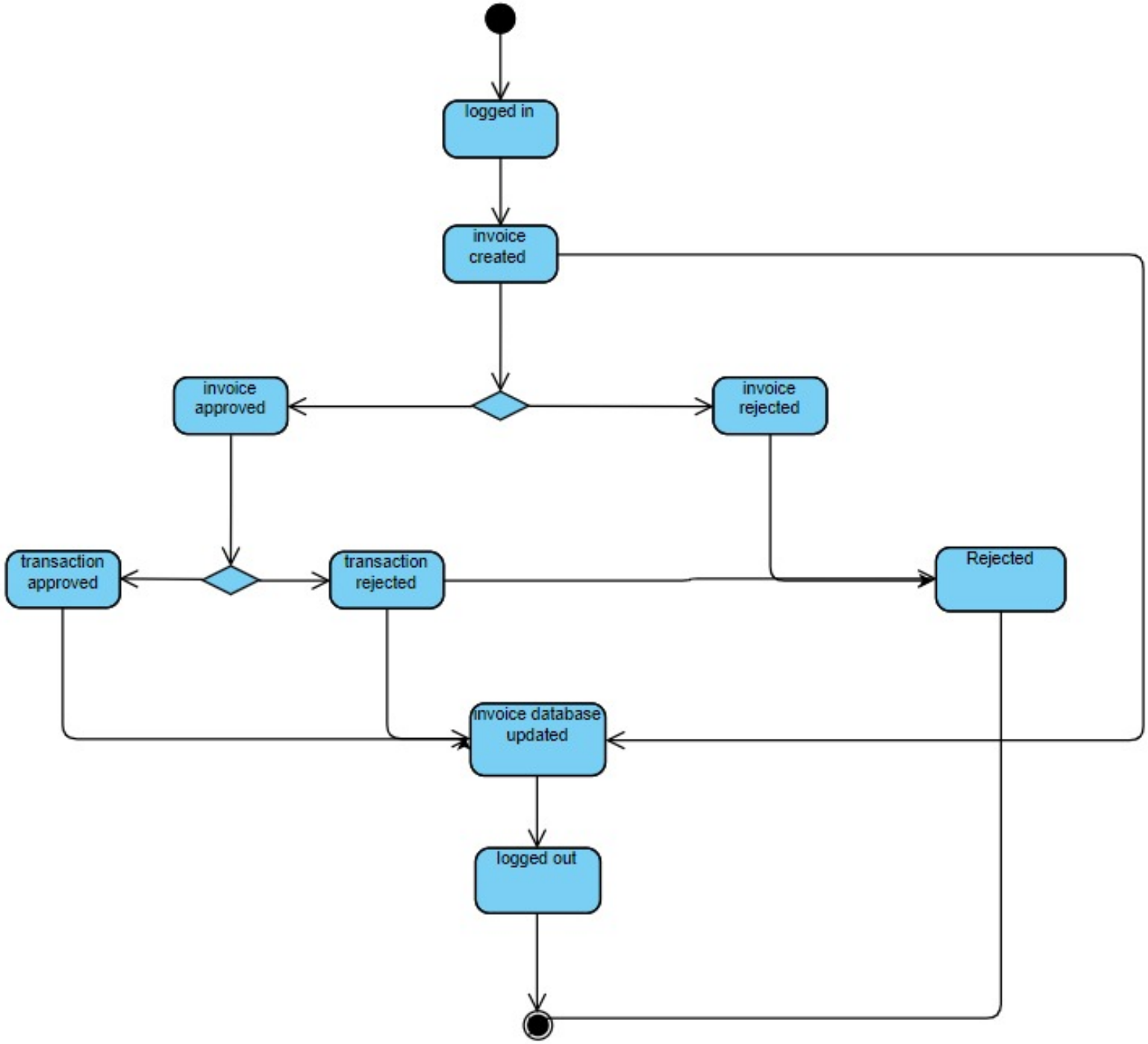

gstno
company_addr
Company
company_name
company_id
company_contact
from
role_id
role
Role
has
belongs to
user_id
company_id
user_role
user
last_name
first_name
contact
creates
auth_id
Authorisation
pwd
user_email
authorises
invoice_id
invoice_state
log_date
invoice_status
has
invoice_id
invoice_no
invoice
invoice_date
invoice_client
invocie_client
Invoice_amt
invoice_vendor
invoice_file

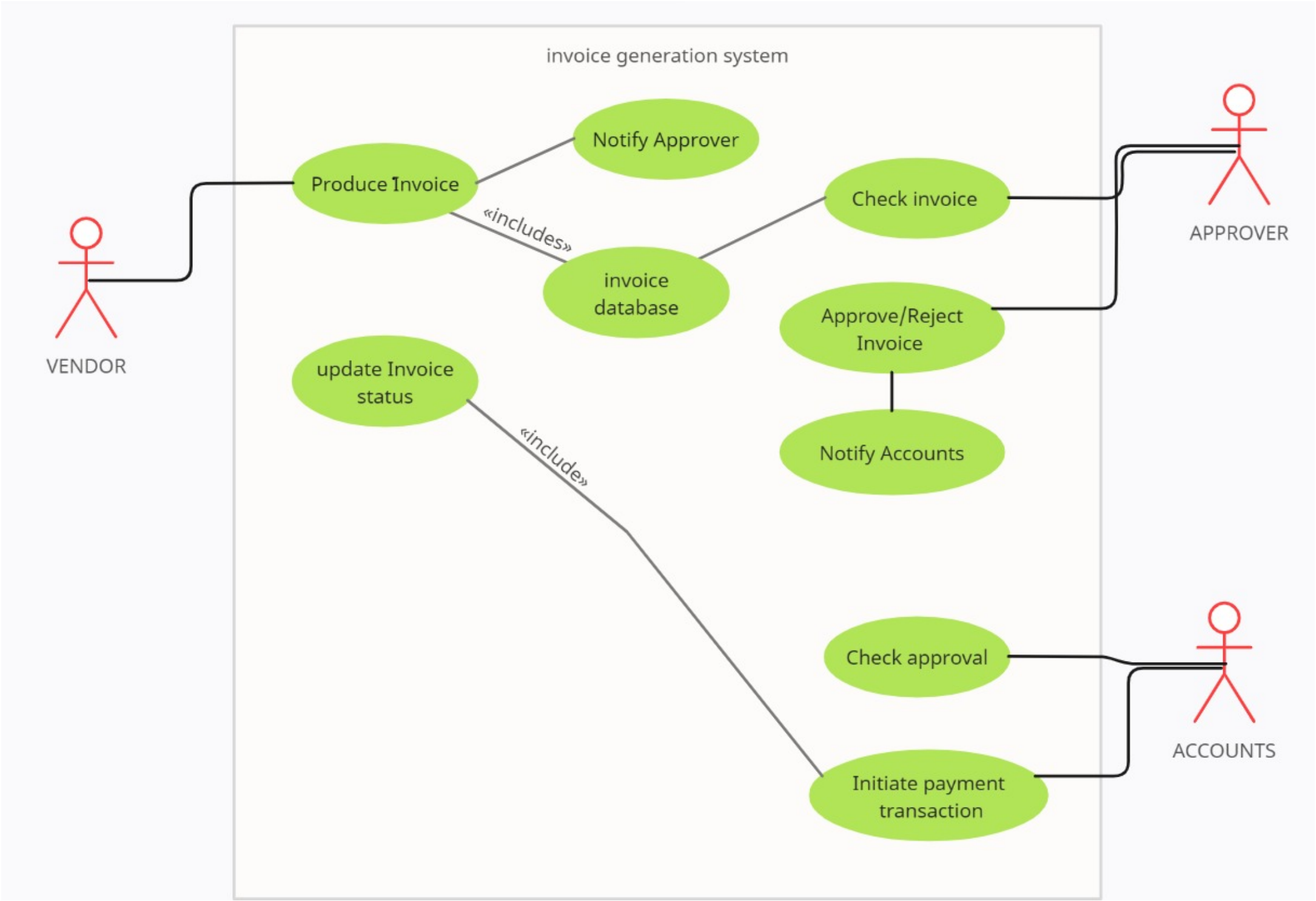

LEVEL 0
APPROVER
approval
invoice
HANDLING INVOICE
invoice
money
VENDOR
money
approval
ACCOUNTS

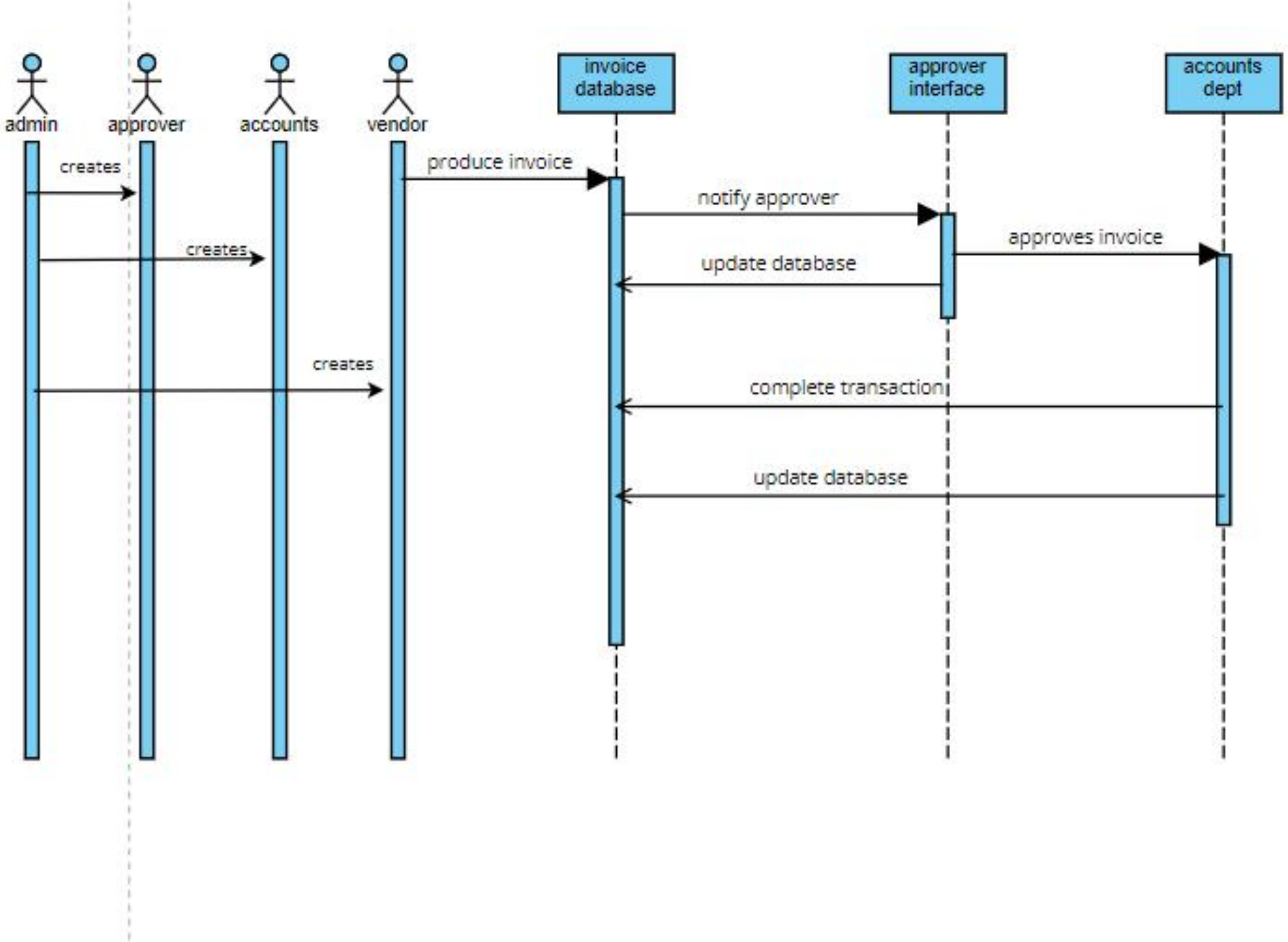

VENDOR
invoice
1
PRODUCE INVOICE
notification
2
NOTIFY APPROVER OF NEW INVOICE
make payments
INVOICES
request details
receive details
4
PAYMENT
APPROVER
ACCOUNTS
notifies of approval
3
APPROVE INVOICE
3
CHECKS DETAILS OF INVOICE

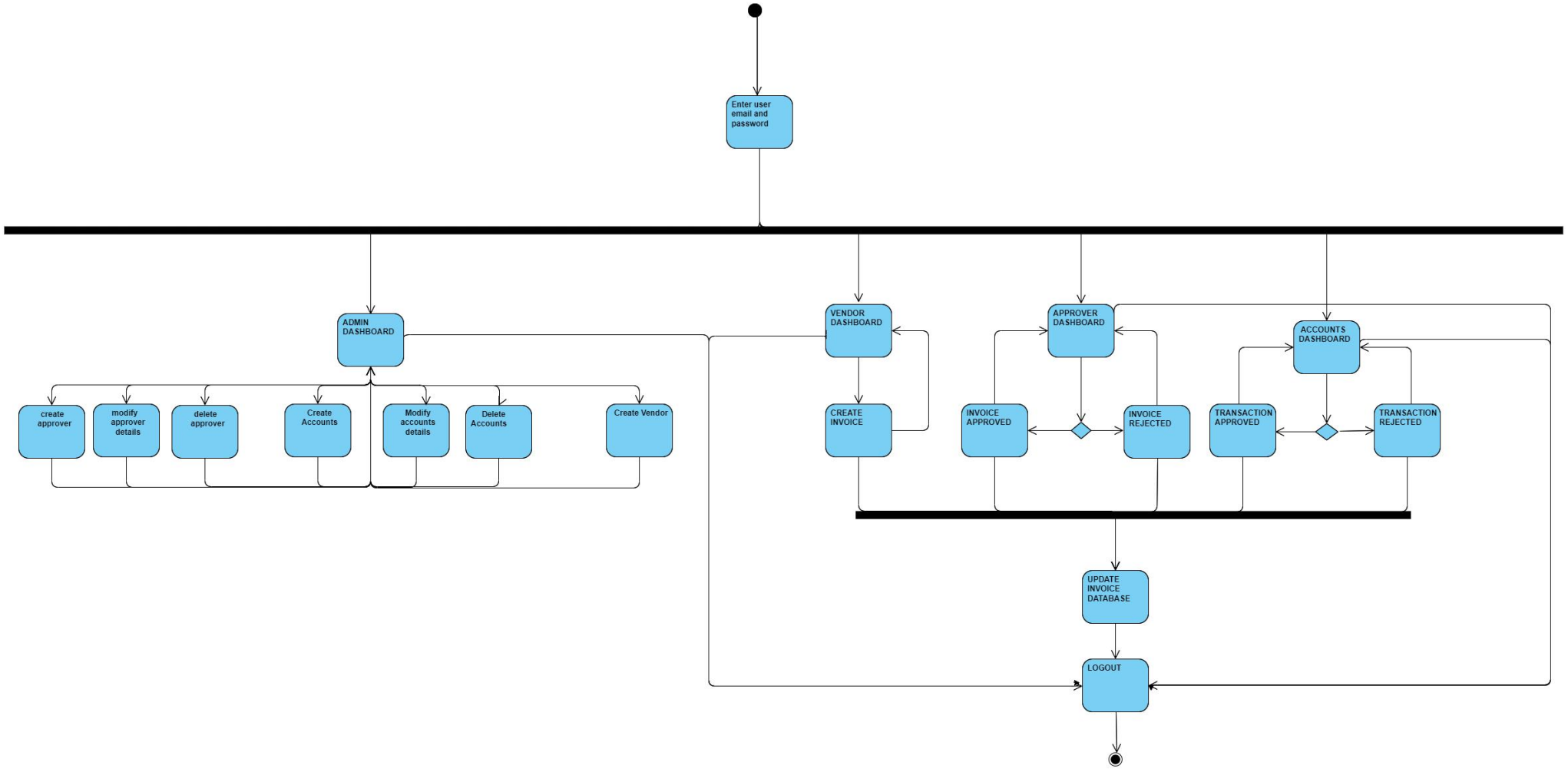

login credentials
login credentials
vendor login
vendor logs in
VENDOR
invoice of order
1
PRODUCE INVOICE
2
NOTIFY APPROVER OF NEW INVOICE
login credentials Database
login credentials
make payments
INVOICES database
request details
notifies the approver of new invoice
4
PAYMENT
receive details
APPROVER
approver logs in
approver login
accounts logs in
ACCOUNTS
notifies of approval
3
APPROVE INVOICE
3
CHECKS DETAILS OF INVOICE
accounts department login
login credentials

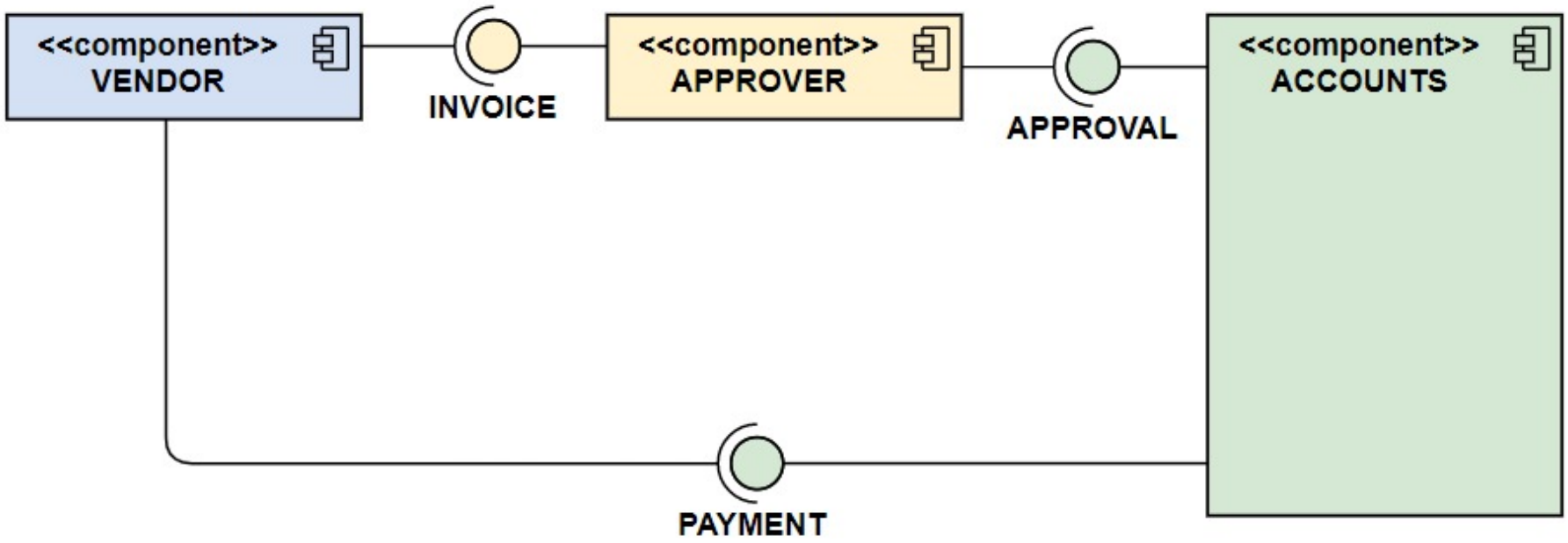

admin
+UserID: int
+First name: string
+Last name: string
+Contact: string
+User role: admin
+ add employee()
+add vendor()
+modify employee details()
+delete employee()
+view all employee()
+view all vendors()
vendor
+invoice date : int
+invoice amount : int
+company id : int
+invoice status: int
+create invoice()
+delete invoice()
+Invoice status()
accounts
+invoice Id: int
+invoice date: int
+invoice amount: int
+company id: int
+invoice state: int
+approve payment()
+reject payment ()
approver
+Invoice Id: int
+ invoice date : int
+Invoice amount : int
+invoiced by company: int
+ approve invoice()
+ reject invoice ()
invoice
Invoice id:int
invoice date: int
invoice amount: int
company id: int
invoice state: int

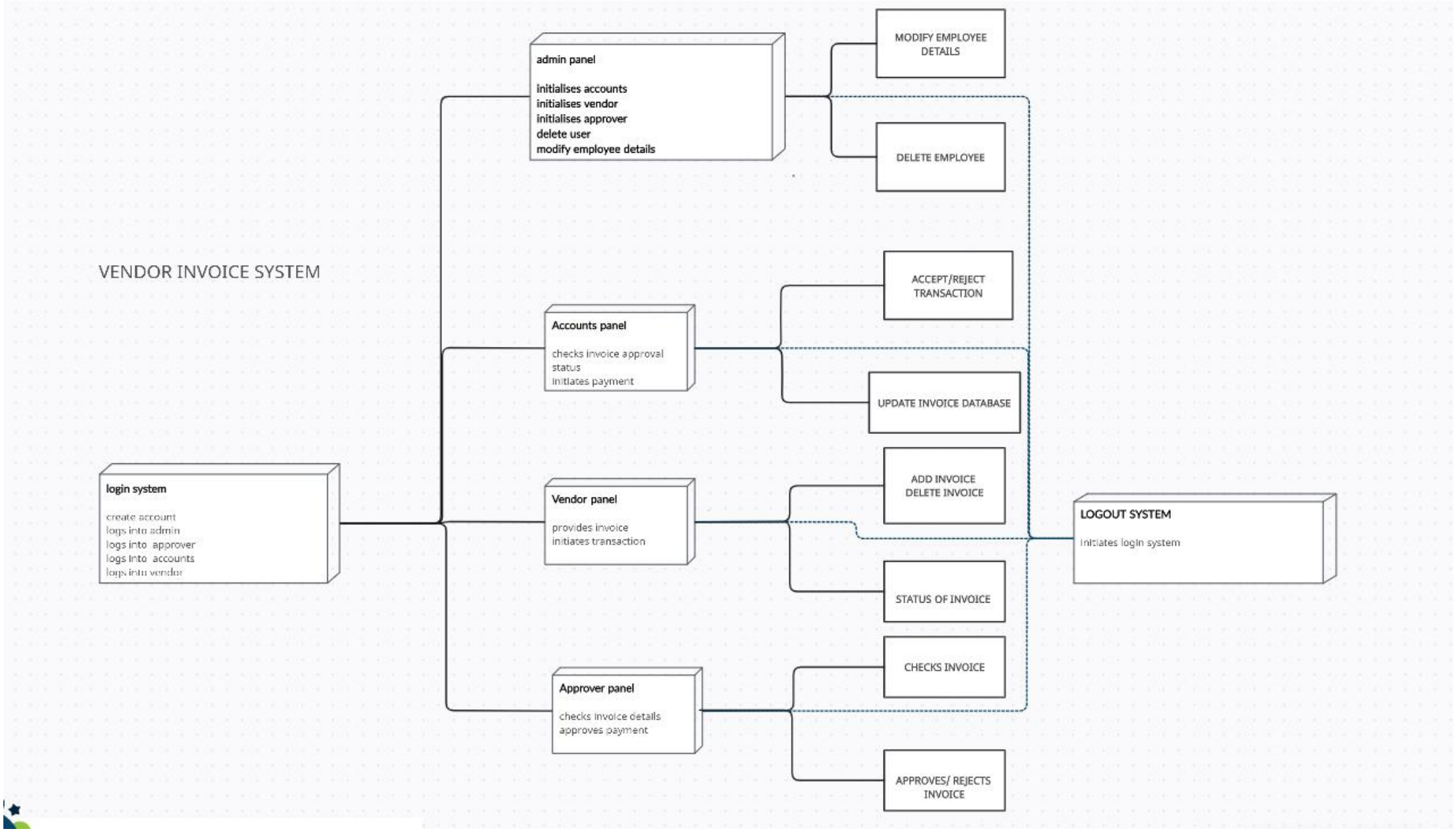

logged in
invoice created
invoice approved
invoice rejected
transaction approved
transaction rejected
Rejected
invoice database updated
logged out


invoice generation system
Notify Approver
Produce Invoice
«includes»
invoice database
Check invoice
APPROVER
VENDOR
Approve/Reject Invoice
update Invoice status
«include»
Notify Accounts
Check approval
ACCOUNTS
Initiate payment transaction


admin
approver
accounts
vendor
invoice database
approver interface
accounts dept
creates
creates
creates
produce invoice
notify approver
approves invoice
update database
complete transaction
update database


Enter user email and password
ADMIN DASHBOARD
create approver
modify approver details
delete approver
Create Accounts
Modify accounts details
Delete Accounts
Create Vendor
VENDOR DASHBOARD
CREATE INVOICE
APPROVER DASHBOARD
INVOICE APPROVED
INVOICE REJECTED
ACCOUNTS DASHBOARD
TRANSACTION APPROVED
TRANSACTION REJECTED
UPDATE INVOICE DATABASE
LOGOUT


<<component>>
VENDOR
INVOICE
<<component>>
APPROVER
APPROVAL
<<component>>
ACCOUNTS
PAYMENT


VENDOR INVOICE SYSTEM
admin panel
initialises accounts
initialises vendor
initialises approver
delete user
modify employee details
MODIFY EMPLOYEE DETAILS
DELETE EMPLOYEE
Accounts panel
checks invoice approval status
initiates payment
ACCEPT/REJECT TRANSACTION
UPDATE INVOICE DATABASE
login system
create account
logs into admin
logs into approver
logs into accounts
logs into vendor
Vendor panel
provides invoice
initiates transaction
ADD INVOICE
DELETE INVOICE
STATUS OF INVOICE
LOGOUT SYSTEM
initiates login system
Approver panel
checks invoice details
approves payment
CHECKS INVOICE
APPROVES/ REJECTS INVOICE