Lab.cafe

Documentation

Benovicsová Lucia, Forgáč Adam, Mihál Lukáš, Vančo Michal

Content

Obsah

A. Catalogue of requirements

1 Introduction

1.1 Purpose of requirements document

This document is intended to provide a detailed overview of the specific requirements of the Lab.cafe project, which is being carried out as part of the course Information Systems Development. The aim of the project is to allow students to apply the theoretical knowledge they have gained during their bachelor's studies at the Faculty of Mathematics, Physics, and Informatics at Comenius University during the years 2022–2025. The requirements catalogue serves as a communication tool among all the stakeholders of the project, ensuring that everyone clearly understands the system's functionality, objectives, and value. This document provides a detailed overview of the project scope, its expected features, user requirements, and technical specifications. This catalogue of requirements serves as a bounding contract between the team and the client.

1.2 Scope of the product

The developed information system is designed to manage access control and order processing for Lab.cafe and its technical infrastructure.

Its core function is to facilitate communication between the Otello system and various devices, such as 3D printers, locks, and access modules. Through integration with the Fabman database, the system enables the administrator to oversee member access to both the workshop and equipment, while also managing and tracking 3D printing tasks.

Furthermore, the system incorporates POS terminal capabilities to monitor café purchases made by members, utilizing card-based authentication and synchronizing data via platforms like PapayaPOS and Airtable. This solution streamlines member management, access control, and the tracking of equipment usage within Lab.cafe.

1.3 Definitions, acronyms and abbreviations

POS – point of sale

1.4 References

GitHub repository of the project: https://github.com/TIS2024-FMFI/Lab.cafe
Source code of the client: https://github.com/LabCafe6.

1.5 Overview of the remainder of the document

The second chapter describes the general functionality of the system, its characteristics, dependencies, constraints placed on it and the perspective of the product as a whole. The third chapter provides a detailed specification of the functional requirements for the system under development as well as those requirements that not directly related to its functionality.

2 General description

2.1 Product perspective

The system serves two primary functions: managing a workshop and a café. Members of the Lab.cafe community use a membership card to access both areas. This membership card acts as an authentication method for various services, such as entering the workshop, operating Lab.cafe equipment(specifically 3D printers), and applying membership benefits like café discounts. The card ensures that members can easily verify their identity and gain access to services without manual intervention. Currently, the system operates in a prototype form but requires optimizations to improve performance, particularly in access speed and control logic.

2.2 Product functions

The system is composed of several key components and integrates various tools and services.

2.2.1 Printer Management and Order Processing

The internal system, Otello, manages the list of printers and processes orders. The system is built on CodeIgniter (PHP, MySQL). Member and machine access data are stored in the Fabman database. Memberships are purchased through WooCommerce (WordPress), with billing handled via the WooCommerce Subscriptions module. Membership information is synchronized with Fabman.iovia Make.com.

2.2.2 Access Control for Workshop

When a member taps their card on the access module at the workshop door, the module sends an endpoint request (including the card number) to a central server on the Otello system. The server retrieves access rights from Fabman, determining the member's membership type. Based on the response, the module signals access approval, denial, or an error through three LED lights. Lab.Cafe uses custom hardware for this access module.

2.2.3 Equipment User Control

The Fabman Bridge controls machine access, verifying if the user via their RFID card. If authorized, the Fabman Bridgepowers on the equipment. Each 3D printer is equipped with a Raspberry Pi Zero2W running OctoPrint, which monitors the printer's status and active print jobs. A central server (running on a Raspberry Pi) manages these printers and their status. The system tracks print jobs and their authors for billing purposes, with monthly consumption calculated based on usage. Printer management and configuration are handled via the Otello system.

2.2.4 Café Discounts

A POS terminal tracks members' café purchases using their membership card. Custom hardware retrieves the member's details and stores the current valid name on the server. This information is retrieved by PapayaPOS during checkout. Once the transaction is processed in PapayaPOS, the account information is saved in Airtable.

The member presents their card at the bar when paying for a meal. The LabPOS terminal verifies the member’s identity and displays their name. The bartender

processes the payment in PapayaPOS, which triggers a call to a predefined endpoint in the Otello system.

2.3 User characteristics

The system distinguishes between two types of users:

Administrator – has full access to the system and can modify or update it as needed.

Regular user – The users of our software will be clients of Lab.cafe, specifically those using their maker space. On the Lab.cafe web shop, Once their membership is set, they freely use the workshop space for tasks like school projects, launching a startup, or experimenting with new ideas.

2.4 General constraints

The system relies on external services such as Fabman, Woocommerce,Make.com, Superfaktura, Airtable, and Octoprint, and any outages or changes to these services can impact its functionality. Hardware devices like RaspberryPi have limited performance, which could cause issues during more demanding operations. Processes such as machine access or claiming a 3D print job require quick responses from users, which may sometimes be impractical. The system's functionality also depends on the reliable performance of API calls that connect various components.

2.5 Assumptions and dependencies

The system's operation assumes the stable performance of external services like WooCommerce, Fabman, Make.com, and others. The internal Otello system must be reliable since it manages critical functions like access to 3D printers and locks. Fabman Bridge and RDIF technology must reliably verify user access. The system requires a stable internet connection to ensure all devices, including Raspberry Pi and terminals, function properly. Additionally, it is crucial that hardware is well-configured and regularly updated to prevent any issues during operation.

3 Specific requirements

3.1 Workshop requirements

a) The central server displays the list rooms.

b) Every printer can be added to one room.

c) Machine usage is only possible after presenting a valid membership card.

d) The central server has an overview of all machines.

e) Each machine is connected to a Raspberry Pi running Octoprint, which monitors the machine's status.

f) The central server displays the list and current status of the machines.

g) The central server records each machine's job on the server, specifically:

- Membership card ID
- Start of machine usage
- End of machine usage• Machine ID
- Job performed (filename)
- Job progress in %
- Job logs

h) Machine configuration is done via the internal Otello system.

i) The central server offers the user the option to acquire the machine via button on the terminal.

j) During active work, the machine is unavailable for starting another job.

k) Once the work is completed, the machine becomes available again for use.

l) The central server updates list of printers every 5 minutes.

3.2 Door access requirements

a) Door opening is only possible with a valid membership card.

b) The membership card is presented to the module.

c) The module indicates the result status with 3 LED lights:

- Green – access granted
- Red – access denied
- Flashing red – an error occurred

d) The module communicates with the Otello system's API.

e) The Otello system determines door access.

f) Access evaluation must be completed within 1 second.

g) Upon successful verification, the doors open for 3 seconds.

i) Access is granted even if the connection to the module fails.

3.3 Café requirements

a) A membership card reader module, LabPOS, is located at the checkout.

b) During payment, the membership card can be scanned.

c) The module will display the member's name on the screen based on the scanned membership card.

d) The module saves the current member's name on the server using an APIcall.

e) After the transaction is completed, the PapayaPOS system saves the member’s bill in Airtable.

B. Design

1 Detailed specification of external interfaces

The machine, access, and member management system at Lab.cafe integrates multiple applications, devices, and file formats to ensure seamless functionality and efficient data management.

1.1 3D Printers

3D printers communicate with the Otello system through the HTTP REST API provided by Octoprint (for detailed information here) Each printer is equipped with a Raspberry Pi Zero 2W running Octoprint, enabling print monitoring and retrieval of active job information. Otello periodically (every 5 seconds) queries the printers to fetch their status. The list of printers is updated every 5 minutes, checking if any new device was added. A centralized database logs all print jobs for monthly summaries and user billing purposes.

1.2 Database Usage

The system utilizes a MySQL database to store printer configurations, print job records, member data, and access rights. Additional data, such as consumption transactions, is synchronized with Airtable.

1.3 Membership and Machine Access

For membership and machine access management, the system integrates with Fabman.io, which uses REST API to verify user access to specific machines. When an RFID card is scanned at a Fabman Bridge device, Fabman checks user permissions and, if authorized, activates the machine for a specified duration.

1.4 Payment Processing

In processing payments at the bar, the system uses PapayaPOS in conjunction with the LabPOS terminal. A member scans their RFID card at the LabPOS terminal, which

calls an endpoint in the Otello system to identify the active member. PapayaPOS then communicates with Otello via a simulated hotel API to retrieve active member data. Upon transaction confirmation, PapayaPOS sends the payment details back to Otello, where they are stored in Airtable.

1.5 Membership Management

Regarding memberships, the system integrates Woocommerce and Fabman.io via Make.com. Memberships purchased or renewed through Woocommerce are automatically synchronized with Fabman.io, ensuring members have access to machines and facilities.

1.6 User Interface

The user interface of the Otello system allows standard users to monitor printer statuses and manage print jobs. Administrators have access to an extended range of functions, including adding new printers, rooms, members, and configurations. Device or member configurations can also be managed in bulk through JSON file uploads (example of json file structure following) . Additionally, the system supports exporting and importing print job data in CSV format.

```
{
  "temperature": {
    "tool0": {
      "actual": 214.8821,
      "target": 220.0,
      "offset": 0
    },
    "tool1": {
      "actual": 25.3,
      "target": null,
      "offset": 0
    },
    "bed": {
      "actual": 50.221,
      "target": 70.0,
      "offset": 5
    },
    "history": [
      {
        "time": 1395651928,
        "tool0": {
          "actual": 214.8821,
          "target": 220.0
        },
        "tool1": {
          "actual": 25.3,
          "target": null
        },
        "bed": {
          "actual": 50.221,
          "target": 70.0
        }
      },
      {
        "time": 1395651926,
        "tool0": {
          "actual": 212.32,
          "target": 220.0
        },
        "tool1": {
          "actual": 25.1,
          "target": null
        },
        "bed": {
          "actual": 49.1123,
          "target": 70.0
        }
      }
    ]
  },
  "sd": {
    "ready": true
  },
  "state": {
    "text": "Operational",
    "flags": {
      "operational": true,
      "paused": false,
      "printing": false,
      "cancelling": false,
      "pausing": false,
      "sdReady": true,
      "error": false,
      "ready": true,
      "closedOrError": false
    }
  }
}
```

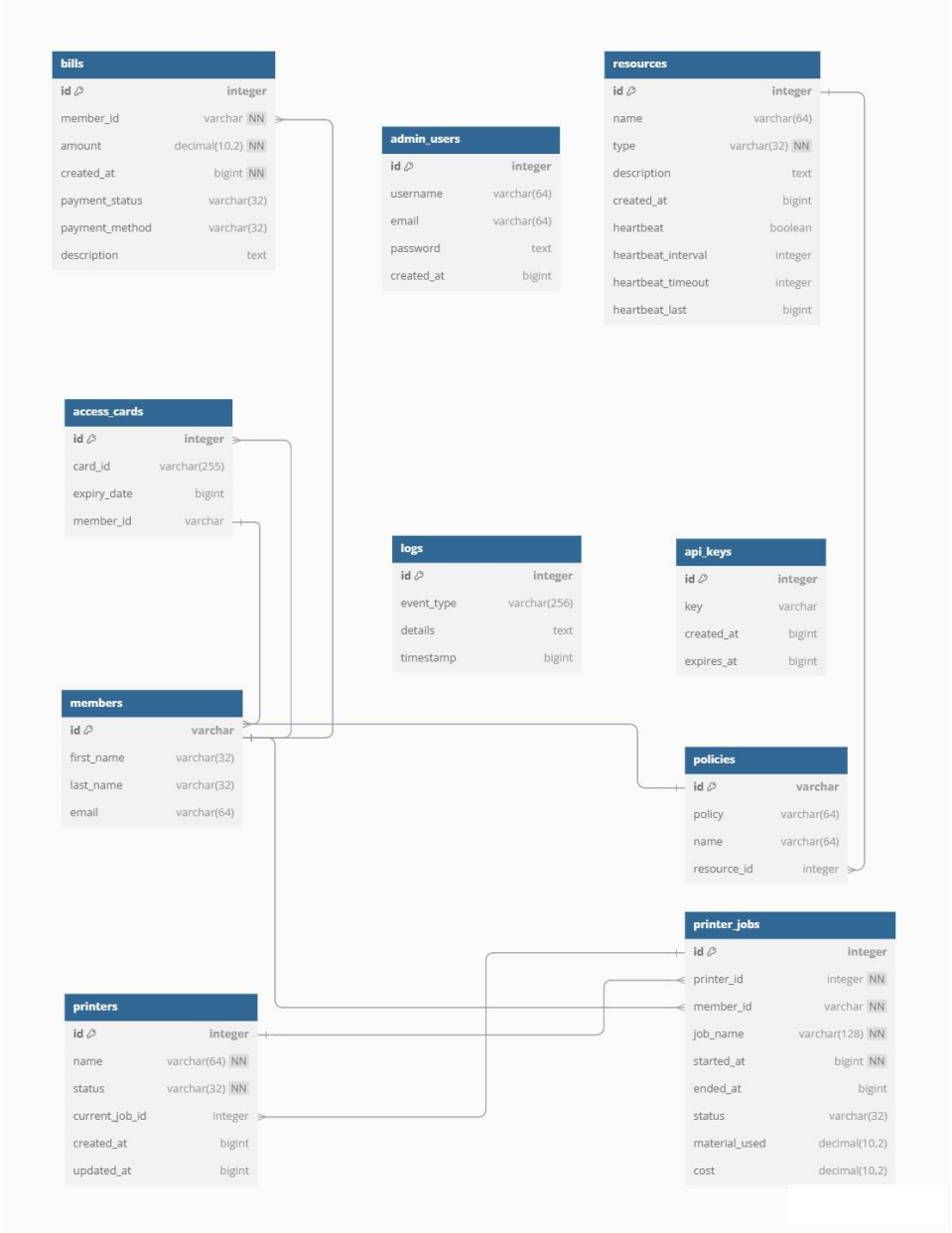
2 Detailed Data Model

The data model represents the system's core entities and their relationships, focusing on managing members, access control, billing, and resource usage.

Access Cards store membership card details linked to Members, who are the primary users of the system. Members can have associated Bills, tracking charges for services like 3D printing. Resources, such as printers or doors, are controlled by Policies, which de ne access and usage rules.

Printers store the status of 3D printers and their current jobs, while Printer Jobs track individual tasks initiated by members, linking printers to specific users.

Admin Users manage the system and indirectly interact with entities such as bills or policies. Logs record system events, offering insights into activities like resource usage or errors. API Keys secure system endpoints, facilitating authenticated communication. This model ensures modularity and scalability, with clear relationships between entities where necessary, while allowing f l exibility for independent tables like logs and api_keys.



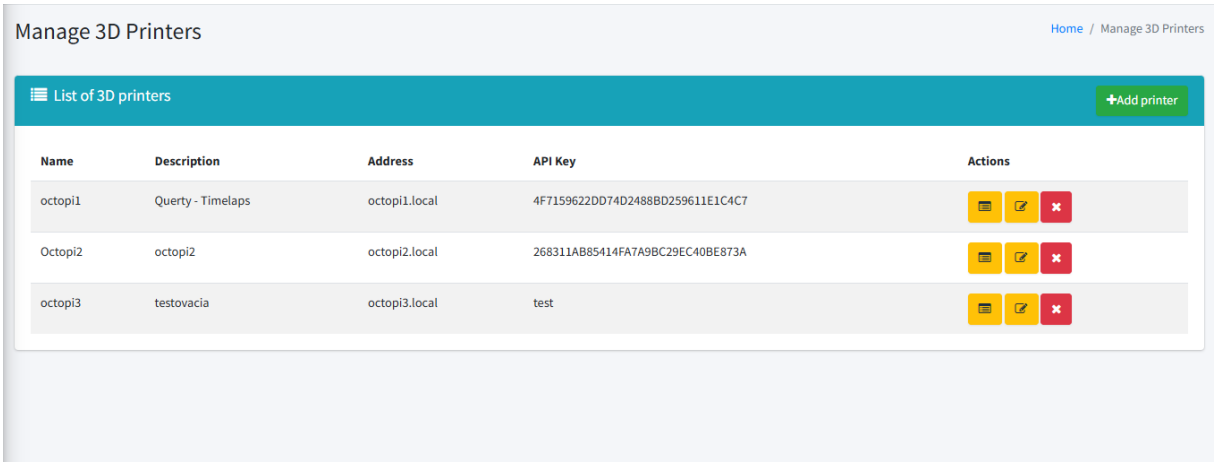
Img. 1: Illustration of the Database Diagram

3 User Interface Design

The graphical user interface (GUI) is designed specifically for managing 3D printers. This interface operates on a standalone microcomputer using PHP, providing an interactive display of all printers retrieved from the Otello system. It periodically fetches the latest printer statuses, ensuring real-time updates and evaluations of their condition.

3.1 Printer List Interface

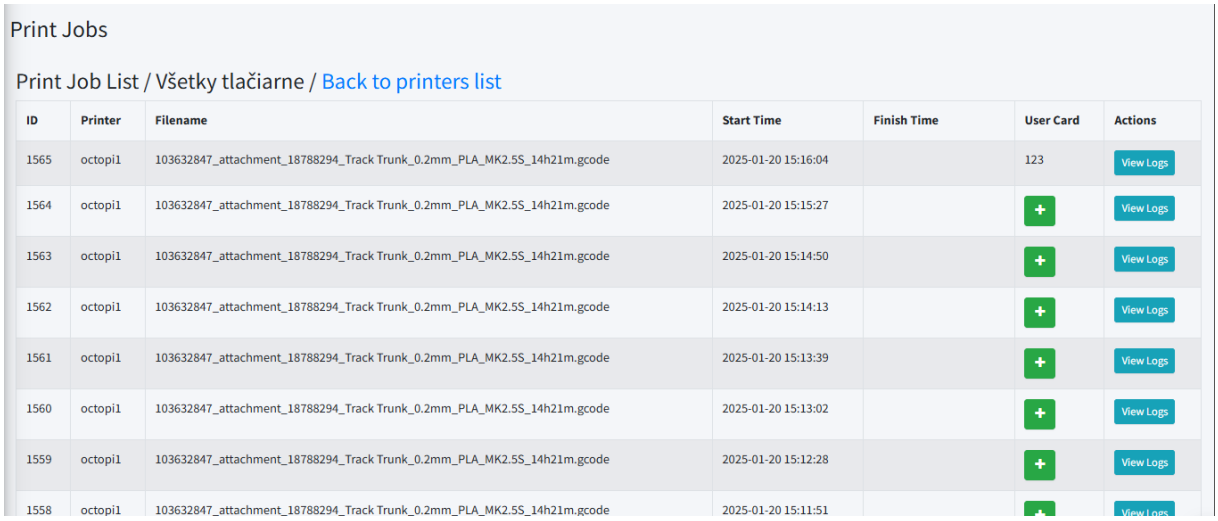
The main interface displays a responsive list of printers. Printers can be added or deleted, and the data shown can be edited.



Img. 2: Illustration Showing Printer's list interface

3.2 Printer Jobs

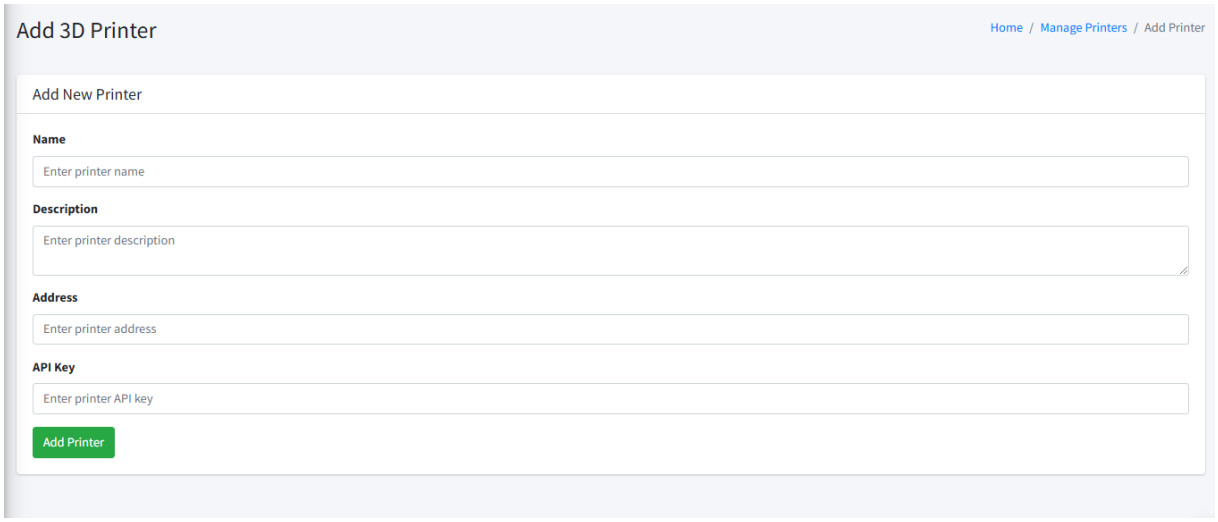
Each printer's job is clearly displayed with its ID, printer name, filename (of what is printing), start and finish time of printing, username – if there is no user it gives ability to add user. Logs of each job can be displayed.



Img. 3: Illustration Showing Print jobs

3.3 Add Printer

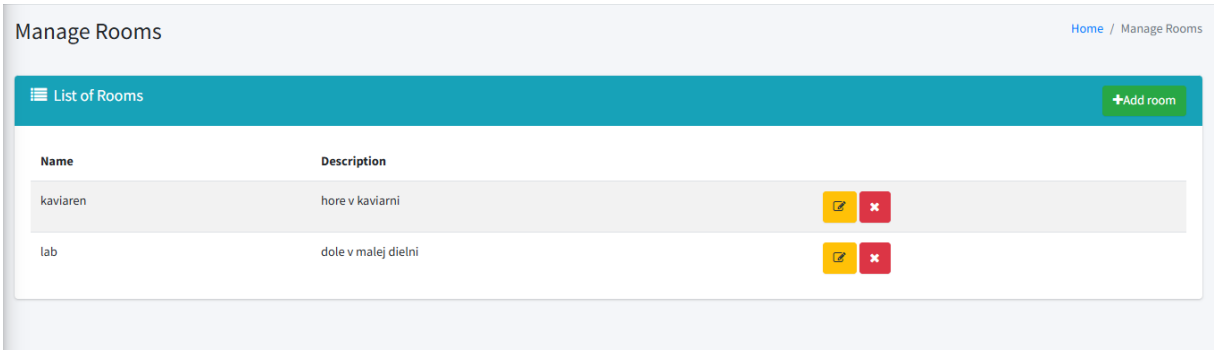
To add new printer, new page displays to type in printer name, description, address, and API key. When successful, display message confirming successful action.



Img. 4: Illustration of Add Printer page

3.4 Rooms List Interface

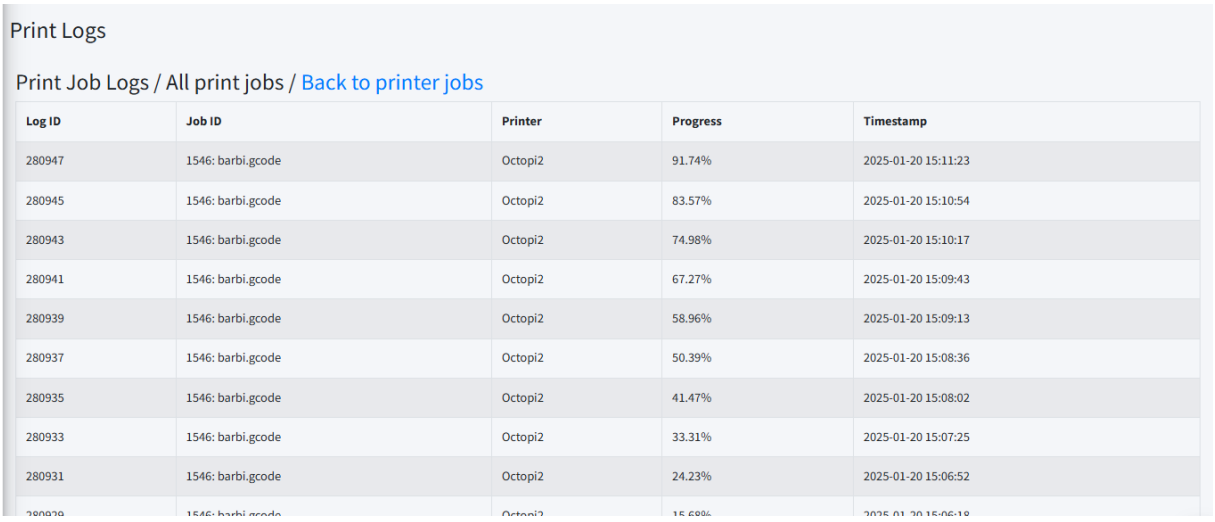
Displays a responsive list of rooms. Rooms data can be edited, new room can be added and also deleted. User can manage the list of printer in every room.



Img. 5: Illustration of Rooms list page

3.5 Print Logs page

Simple, easy to read list of all print logs (showing most important data as progress) for print jobs.



Img. 6: Illustration of Print Logs

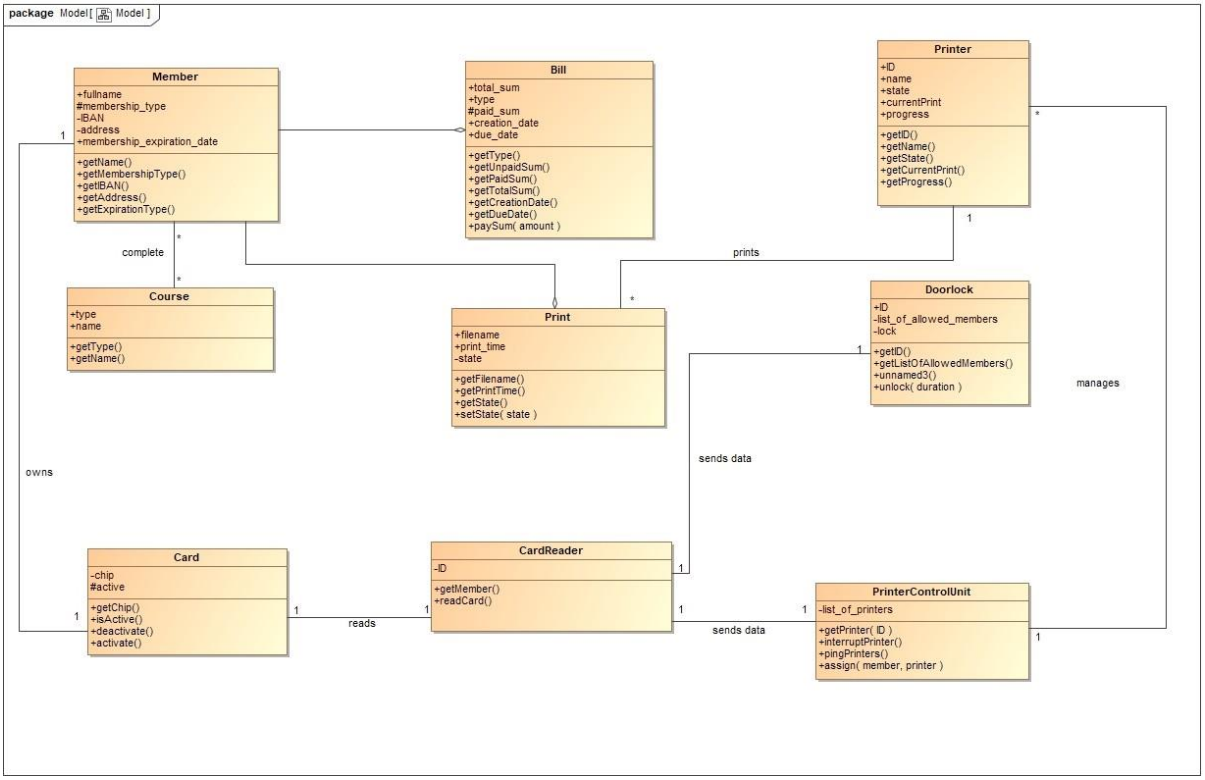
4 Implementation Design

4.1 Class Diagram

This UML class diagram represents a system for managing members, printing services, billing, and access control. The Member class stores user details, membership types, and expiration information. Members can enroll in Courses, receive Bills for services like printing, and access secure areas via Cards.

CardReaders read cards to verify members, while Doorlocks grant access based on permissions. The printing system includes Print jobs, which are managed by Printers under the control of a PrinterControlUnit, responsible for job assignment, printer monitoring, and interruptions. Printers process jobs and track their progress, and Bills handle payments for services used.

Overall, the system ensures secure access, organized printing workflows, and efficient member management for environments like labs, libraries, or co-working spaces.



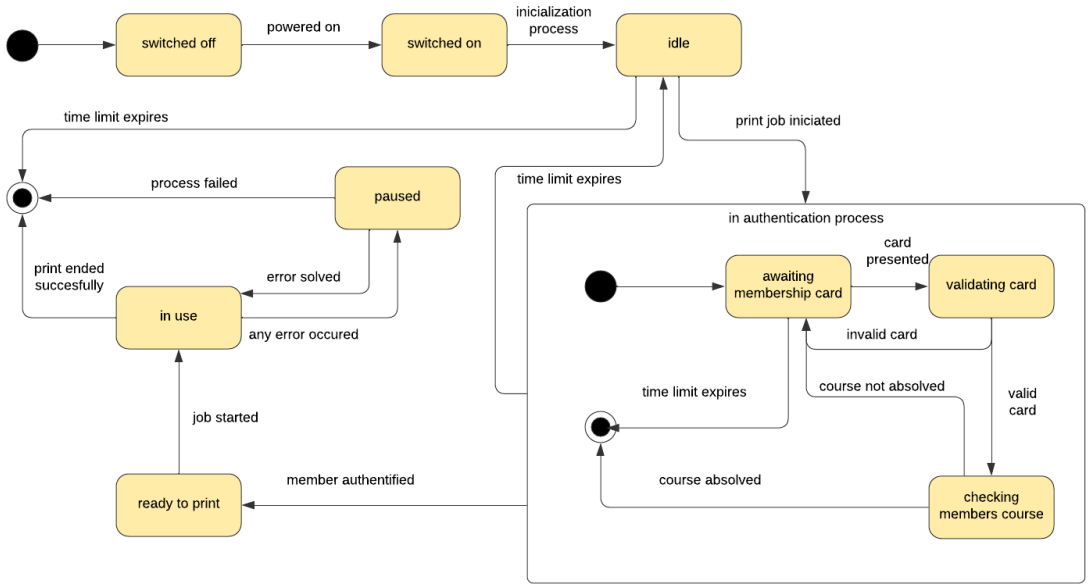
Img. 8: UML Class Diagram

4.2 State Diagram

This state diagram illustrates the lifecycle of a 3D printer, covering power states, user authentication, printing, and error handling.

The printer starts in the switched off state and transitions to idle after powering on and initializing. When a print job is initiated, it enters the authentication process, where the user must present a valid membership card. Upon validation and prerequisite checks, the printer moves to ready to print and then in use during the job.

If the print completes successfully, it returns to idle. Errors move it to a paused state until resolved, after which it resumes printing or returns to idle. If time limits expire or validation fails, the process ends in a terminal state, ensuring secure and the client operation.



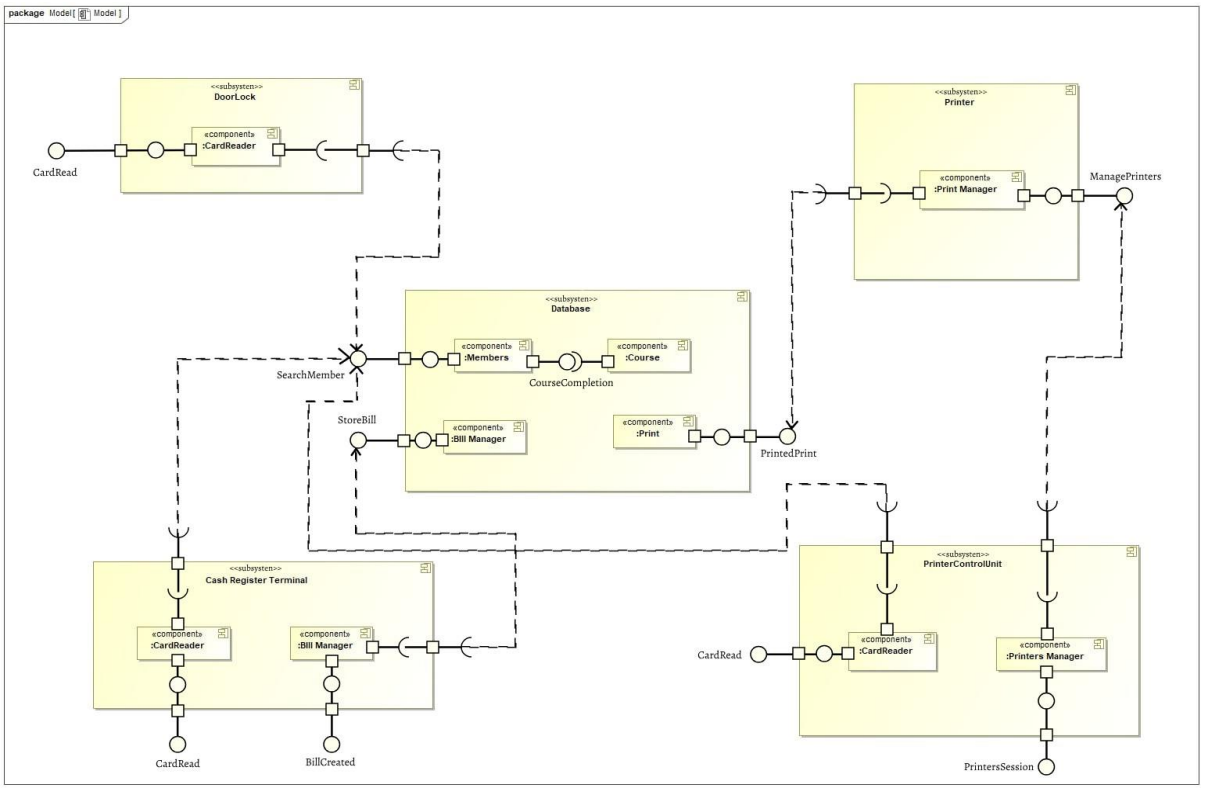
Img. 9: State Diagram of a 3D Printer

4.3 Component Diagram

This component diagram depicts the architecture of a system managing printing services, billing, and access control. The system includes several subsystems, such as DoorLock, Database, Printer, Cash Register Terminal, and PrinterControlUnit, each containing specific components.

The DoorLock subsystem uses a CardReader to authenticate user access, while the Database manages key components like Members, Courses, Bills, and Print jobs, ensuring user data, course completions, and billing are stored and retrieved. The Printer subsystem contains a Print Manager for handling printing operations. The Cash Register Terminal includes a CardReader and a Bill Manager for payment processing.

Finally, the PrinterControlUnit coordinates printers via its Printers Manager and handles user sessions through its CardReader. Interactions between subsystems are marked by dashed connectors, showing workflows like card reading, user validation, course completion checks, billing, and print job processing. This architecture ensures secure user access, efficient printing, and proper billing in an integrated environment.



Img. 10: Component Diagram of the System Architecture

5 Implementation Plan

5.1 External Interfaces

5.1.1 3D Printers

Setup Octoprint: Install Octoprint on Raspberry for each 3D printer.

API Integration: Develop HTTP REST API endpoints for communication between printers and the Otello system.

Status Monitoring: Implement periodic status checks and job claiming mechanism.

5.1.2 Database Usage

Database Setup: Configure MySQL database for storing printer configurations, print job records, member data, and access rights.

Data Synchronization: Integrate Airtable for additional data synchronization.

5.1.3 Membership and Machine Access

Fabman.io Integration: Use REST API to verify user access to machines via RFID card scans.

Access Control: Implement access control mechanisms and duration settings for machine usage.

5.1.4 Payment Processing

PapayaPOS Integration: Develop endpoints for communication between Lab-POS terminal and Otello system.

Transaction Handling: Implement payment processing and data storage in Airtable.

5.1.5 Membership Management

Woocommerce Integration: Synchronize memberships purchased or renewed th-

rough Woocommerce with Fabman.io via Make.com.

5.1.6 User Interface

UI Development: Design and develop user interfaces for monitoring printer statuses and managing print jobs.

5.2 Implementation Design

Member Management: Implement classes for managing members, memberships, and access control.

Printing Services: Develop classes for handling print jobs, printer control, and billing.

Printer Lifecycle: Implement state transitions for printer power states, user authentication, printing, and error handling.

Subsystems: Develop components for DoorLock, Database, Printer, Cash Register Terminal, and Printer Control Unit.

Interactions: De ne interactions between subsystems for card reading, user validation, billing, and print job processing.

5.3 User Interface Design

5.3.1 Printer List Interface

Responsive Design: Create a responsive list of printers with dynamic adjustments based on screen width.

Status Display: Implement clear status displays for each printer.

5.3.2 Printer Statuses

State Representation: Develop visual representations for printer states such as Printing, Paused, Finished, and Operational.

5.3.3 Adding a printer

User Interaction: Implement button interactions and give admin abilities such as adding printers.

5.3.4 Error Handling and Feedback

User Feedback: Provide clear messages and feedback for system states and errors.

C. Testing Scenarios

1. Workshop testing scenarios

a. The user pauses the product print and decides to continue.

1.
Action: The user searches for their print.
Result: The octoprint system displays the selected print with an option to pause it.

2.
Action: The user decides to pause the print and presses the "PAUSE" button.
Result: The monitor shows the printer status as "stopped" and provides options for "CONTINUE" to resume and "CANCEL" to cancel the print.
3.
Action: The user wants to continue printing and presses the "CONTINUE" button.
Result: The system resumes printing and changes the printer status to "printing."
4.
Action: The user leaves the monitor without further intervention.
Result: Printing continues uninterrupted until completion. The job finish time is recorded.

b. The user pauses the product print and decides to cancel it.

1.
Action: The user searches for their print job.
Result: The system displays the selected print job with an option to pause it.
2.
Action: The user decides to pause the print and presses the "PAUSE" button.
Result: The monitor shows the printer status as "stopped" and provides options for "CONTINUE" to resume and "CANCEL" to cancel the print.
3.
Action: The user decides to cancel the print and presses the "CANCEL" button.
Result: The system cancels the product print and locks the printer until the print area is cleared.
4.
Action: The user goes to the printer and removes the cancelled product.

Result: The system unlocks the printer, which is now ready for new prints.

5.

Action: The user starts a new print.

Result: The system creates new print job without any reference to the cancelled print.

c. The system checks for available printers, but none are found.

1.

Action: The user logs into the system.

Result: The system authenticates the user and proceeds to check for available printers.

2.

Action: The system completes the search for available printers.

Result: The system displays the message "No printers found."

d. The system checks for available printers, and one is found.

1.

Action: The user logs into the system.

Result: The system authenticates the user and proceeds to check for available printers.

2.

Action: The system finds printers and displays the list.

Result: The user selects a printer from the list to start the print job.

3.

Action: The user monitors the status of their print on the system interface.

Result: The system displays the print status, including details such as:

1. Printer name
2. Printing progress percentage
3. Estimated printing time
4. Printing start time
5. User card id

4.

Action: The printer completes the print job.

Result: The system confirms the print is completed successfully and updates the printer's status to "Operational."

2. Door access testing scenarios

a. Access granted for a regular user with a card

1.
Action: The user presents a valid membership card to the module.
Result: The green LED lights up, indicating access is granted.
2.
Action: The user observes the green light and opens the door.
Result: The door opens for 3 seconds.
3.
Action: The user presents the card, and the system processes the request.
Result: The process completes within 1 second of card presentation.

b. Access denied for invalid card (card id is not in the system e.g. due to membership expiration)

1.
Action: The user presents a membership card to the module.
Result: The red LED lights up, indicating access is denied.
2.
Action: The user attempts to open the door.
Result: The door remains locked.
3.
Action: The user presents the card, and the system processes the request.
Result: The process completes within 1 second of card presentation.

c. Access granted when the Otello system is offline

1.
Action: The system simulates an offline state by disconnecting the network.
Result: The system operates in offline mode, unable to communicate with Otello.
2.
Action: The user presents a valid membership card to the module.
Result: The green LED lights up, indicating access is granted.
3.

Action: The user observes the green light and opens the door.

Result: The door opens for 3 seconds.

4.

Action: The system processes the card request without requiring Otello.

Result: The process completes successfully in offline mode.

3. Café testing scenarios

a. Valid Membership Card

1.

Action: The LabPOS module scans the membership card tapped on the terminal.

Result: The card data is read successfully.

2.

Action: The module sends a request to the database via an API to validate the membership card.

Result: The system confirms that the card is valid.

3.

Action: The system retrieves the customer’s name and membership details.

Result: The member’s name and active status are fetched.

4.

Action: The module displays the member’s name and membership status on the screen.

Result: The waiter sees the confirmation that the customer is a valid member.

5.

Action: The waiter manually applies membership benefits in the payment system.

Result: The transaction is completed with the appropriate membership benefits applied, and the member's details are saved.

b. Expired Membership

1

Action: The LabPOS module scans the membership card tapped on the terminal.

Result: The card data is read successfully.

2.

Action: The module sends a request to the database via an API to validate the membership card.

Result: The system identifies that the membership has expired.

3.

Action: The system retrieves the customer’s name and expired membership status.

Result: The name is fetched, but the membership status shows expired.

4.

Action: The module displays the message "Membership expired. No benefits applied."

Result: The waiter is notified that the membership is no longer valid.

5.

Action: The waiter completes the payment in the system without applying membership benefits.

Result: The transaction is completed without membership benefits, and the expired status is recorded.

c. Invalid Membership Card

1.

Action: The LabPOS module scans the membership card tapped on the terminal.

Result: The card data is read successfully.

2.

Action: The module sends a request to the database via an API to validate the membership card.

Result: The system cannot find a match for the card in the database.

3.

Action: The system returns an error indicating an invalid card.

Result: The card is flagged as invalid.

4.

Action: The module displays an error message: "Invalid card. Please try again."

Result: The waiter is notified that the card is not valid.

5.

Action: The waiter completes the payment in the system without applying membership benefits.

Result: The transaction is completed without membership benefits.