Lost And Found Application

# **Criterion A**

## **Define the Problem**

The continuous problem of ineffective lost item management has surfaced in the busy and dynamic contexts of educational institutions. The lack of an organized strategy for monitoring and managing misplaced objects has clouded the smooth operation of educational institutions, leading to postponed settlements and a higher likelihood of irreversible damage to people's priceless possessions. This disorganized situation highlights the urgent need for a thorough and well-planned solution that goes beyond the constraints of the current framework.

The complexity of lost item management in schools nowadays is a problem that spreads throughout the system and undermines the effectiveness of education as a whole. Staff and students experience extended periods of uncertainty when missing things are not swiftly addressed due to the absence of a centralized and systematic strategy. In addition to lowering the likelihood of finding misplaced items, this inefficiency makes the school community feel irritated and inconvenienced.

Moreover, the difficulties encountered by individuals in charge of missing objects are made worse by the lack of a well-organized system. Prefects, who are responsible for supervising the resolution process, have trouble classifying, monitoring, and efficiently conveying the whereabouts of misplaced objects. In addition to taxing prefects' administrative skills, this disarray affects how the school is seen for its overall capacity to maintain a safe and orderly environment.

The possible loss of priceless possessions emphasizes the need for an all-encompassing solution even more. In addition to undermining people's financial investments, the present inefficiencies in tracking and resolving lost items cause emotional distress for students and their families. The absence of a simplified procedure puts the prompt recovery of objects in jeopardy and adds to the community's sense of insecurity and mistrust.

## Given these difficulties, it becomes clear that the Lost and Found app is an essential project to transform the way lost items are now managed in educational settings. By offering a consolidated, user-friendly, and dynamic platform, it aims to address the various problems caused by inefficiency, disorganization, and the possibility of irreversible loss. By redefining the standards for lost item management, the proposed solution hopes to give the school community a sense of trust, security, and dependability.

## **Rationale for the Proposed Solution**

The proposed Lost and Found application is the result of a methodical process that explores the fundamental problems deeply embedded in the current system. Based on inquiry, diversity, and simplicity, the development gives priority to a comprehensive understanding of the problems at hand.

* Efficiency in Lost Item Management: At the forefront, the primary objective is to establish an agile system that facilitates prompt reporting and effective management of lost items. This design ethos ensures a seamless workflow catering to the needs of both students and prefects, promoting overall system efficiency.
* Role-Based Access: The strategic implementation of distinct roles for students and prefects is a pivotal aspect of the proposed solution. By endowing prefects with enhanced privileges, including database management, item categorization, and marking found items, responsibilities are distributed effectively. Simultaneously, students experience a user-friendly process for reporting lost items.
* Dynamic Item Categorization: A groundbreaking feature involves the application's utilization of dynamic classes for item categories. This innovation introduces flexibility in managing various types of lost items, dynamically deriving subcategories and sub-classes. This adaptability ensures readiness for future changes in categorization requirements.
* Simplicity: The design philosophy revolves around simplicity, ensuring that users can navigate the system effortlessly. Intuitive interfaces and clear workflows contribute to a user-friendly experience, underlining the importance of an accessible and uncomplicated design.
* Diversity: A key strength lies in the application's ability to accommodate a diverse array of lost items through dynamic categorization. This adaptability enhances the system's capability to effectively handle a broad spectrum of lost belongings, aligning with the varied nature of items typically encountered in a lost and found scenario.
* Exploration: Fostering a spirit of exploration and expansion, the proposed solution embraces the dynamic creation of classes. This feature allows for continuous exploration into new item categories and subcategories, promoting adaptability and fostering sustained growth.

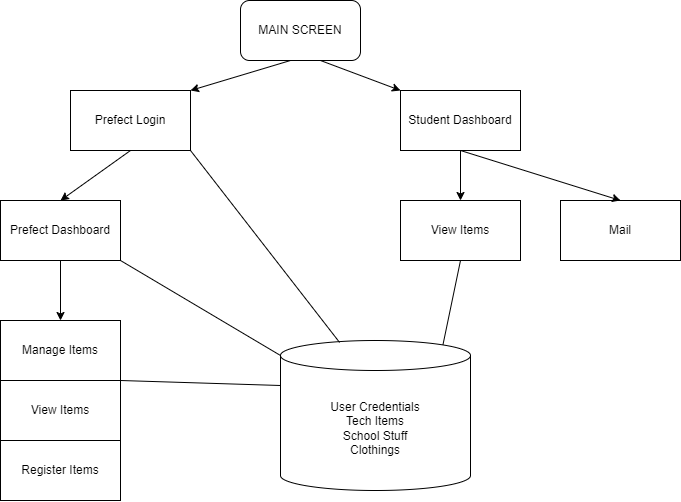
## Using a Python library called Tkinter to integrate a Graphical User Interface (GUI) improves user engagement and makes dashboard navigation and lost item reporting easier. Structured data storage is ensured by Pyodbc's smooth integration of a SQL Server database. Maintaining tables such as UserCredentials and LostItems\_{category\_name} requires the use of SQL queries, which are essential for the application to stay simple and match the developer's skill set for efficient front-end and back-end interaction.

## **Success Criteria**

* User Authentication: Users can securely log in with their email and password, ensuring data protection. The authentication process prioritizes user security while maintaining a seamless login experience for efficient access to the system.
* Efficient Item Registration: Students can efficiently report lost items by providing essential details like email, item description, and subcategory. This streamlined reporting process ensures that all necessary information is captured, laying the groundwork for effective item management within the system.
* Role-Specific Dashboards: Upon login, users are directed to role-specific dashboards, optimizing navigation. Students access features for reporting lost items, while prefects enjoy comprehensive management capabilities. This tailored approach enhances the overall user experience, fostering intuitive interaction with the application.
* Prefect's Management Capabilities: Prefects wield powerful management capabilities, registering lost items, categorizing them by subcategories, and deleting found items. This control empowers prefects to oversee the entire lost and found system, contributing to a more organized and responsive management process.
* Dynamic Item Categorization: The system dynamically creates tables for each category and subcategory, allowing efficient data organization. This adaptive approach ensures the system can accommodate new categories and subcategories seamlessly, offering a scalable and future-proof solution for item categorization.

# **Criterion B**

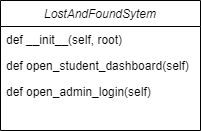
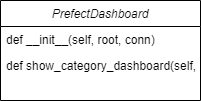
## **Screens:**

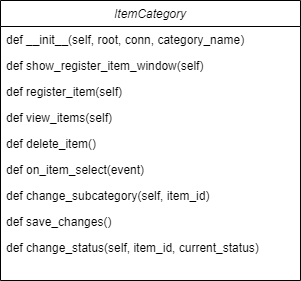


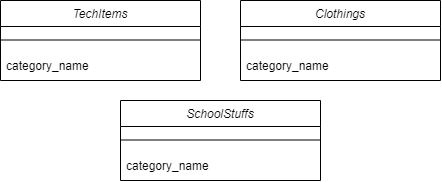
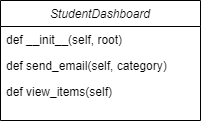
## **Functionality of Each Screen:**

* **Main Screen:** Ask the users, whether they are Students or Prefects.
* **Prefect Login:** Ask the prefect to enter credentials or allows students to go back.
* **Student Dashboard:** Displays student with options like report item and View items list
* **Prefect Dashboard:** Allows the prefect to choose between options like Register new Item or Manage older ones.

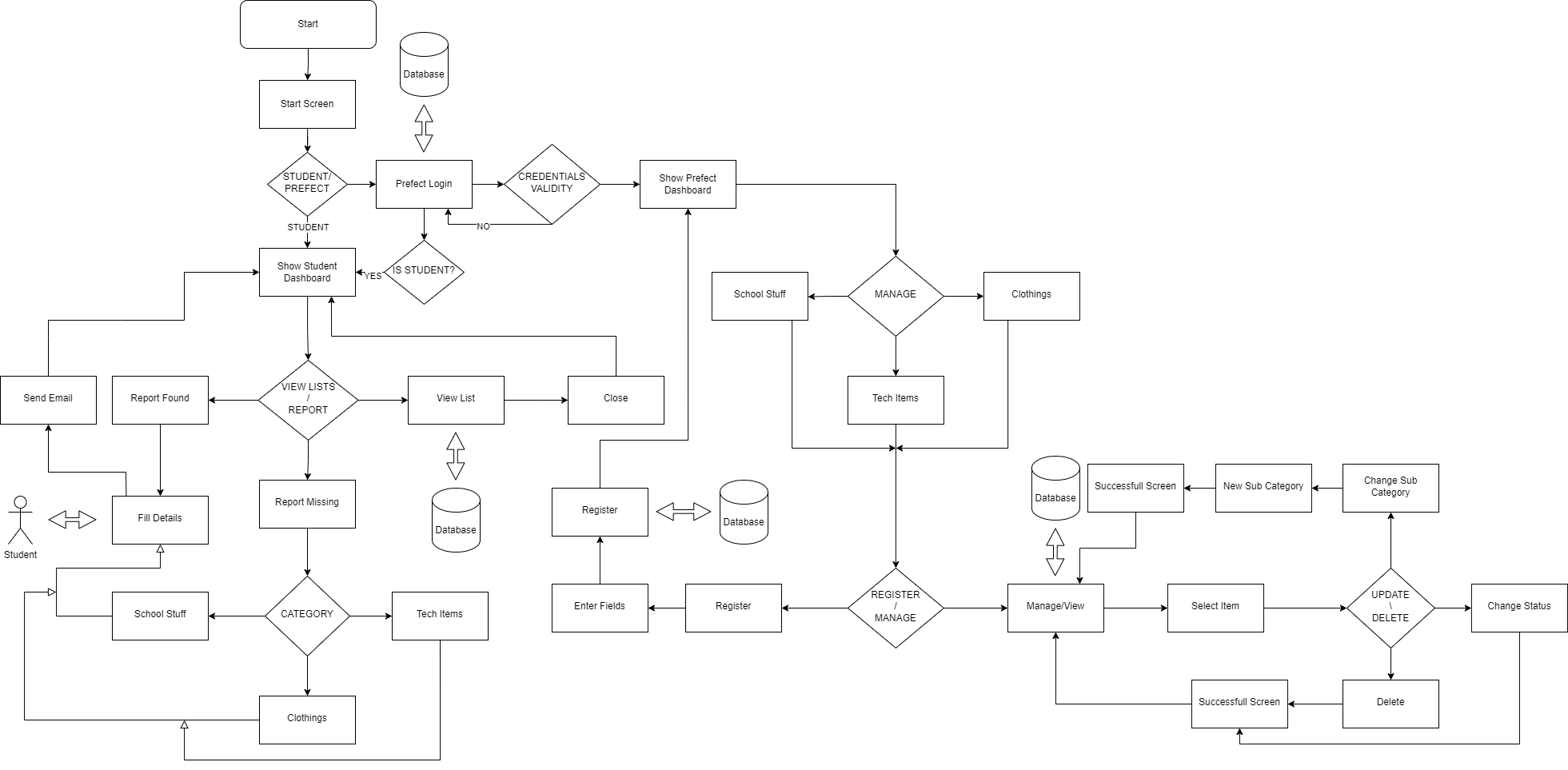
## **Classes UML Diagrams:**



## **Flowcharts:**



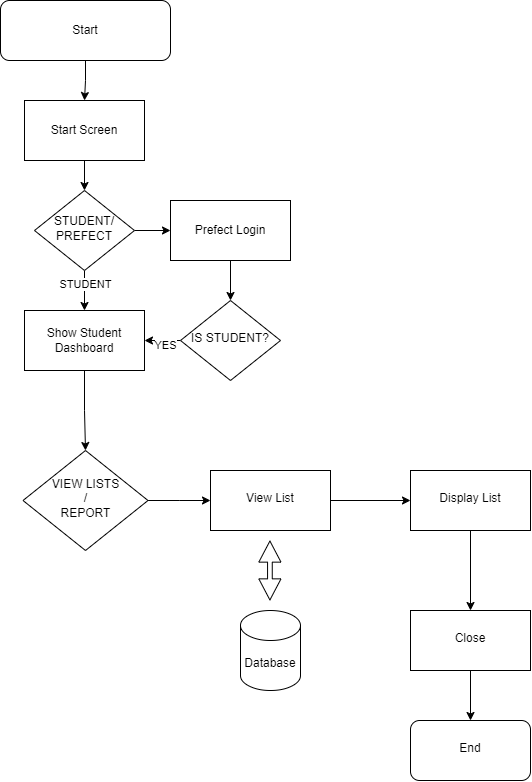


Figure (A)

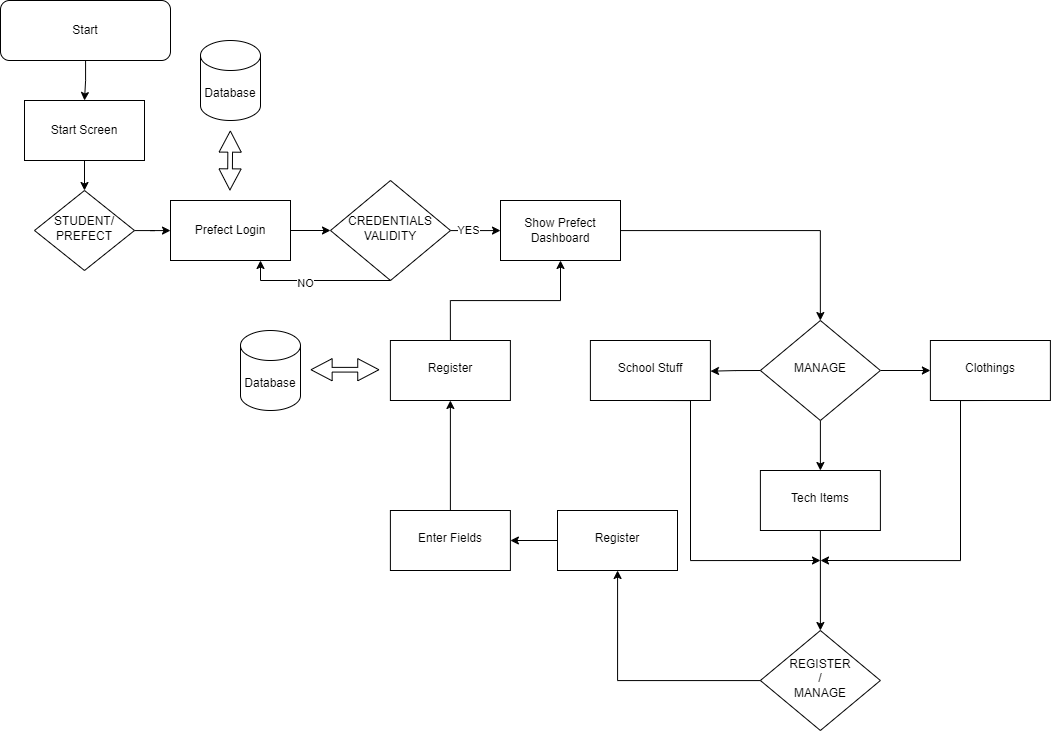


Figure (B)

**A** = How a user can View List of all the lost items registered.

**B** = How a Prefect can register a new item.

## **Database Tables:**

User Credentials:

|  |  |  |  |
| --- | --- | --- | --- |
| **Id** | **Email** | **Password** | **UserRole** |
| 1 | [admin@example.com](mailto:admin@example.com) | Admin123 | Prefect |

TechItems:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Id** | **Owner** | **Status** | **Type** | **Date\_lost** |
| 1 | Maoz | Missing | Laptop | 2024-02-21 |
| 2 | Kamal | Missing | Phone | 2023-12-22 |

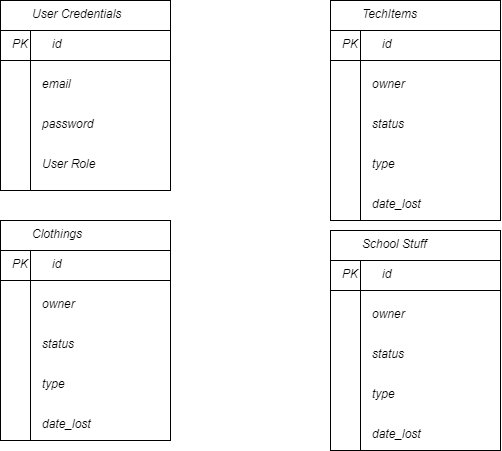
Clothings:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Id** | **Owner** | **Status** | **Type** | **Date\_lost** |
| 1 | Maoz | Missing | Shorts | 2024-02-12 |
| 2 | Kamal | Missing | Socks | 2024-02-12 |

School Stuff:

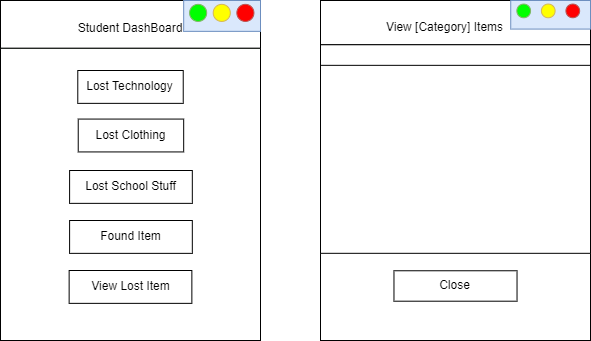
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Id** | **Owner** | **Status** | **Type** | **Date\_lost** |
| 1 | Maoz | Missing | Pencil Box | 2023-12-12 |
| 2 | Kamal | Missing | Calculator | 2024-01-22 |

## **Tables Relation:**



## **Design of GUI:**





## **Record of Tasks**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Task**  **Number** | **Planned action** | **Planned outcome** | **Time**  **estimate** | **Target completion date** | **Criterion** |
|  | Initial Discussion with supervisor (Computer  Science teacher) | This was after having small talks about possible topics for discussion, which eventually led me to my idea that got approval from my supervisor in twenty minutes. | After twenty minutes | 14th May 2023 | A |
|  | Consultation with the client | Understood what the client was struggling with and outlined the appropriate solution. | 10 minutes | 14th May 2023 | A |
|  | Discussion with the Computer Science teacher to select software. | JavaScript, CSS, HTML, PHP, MySQL are chosen as programming languages since I am most comfortable with using those, and they will allow me to design a complex product to satisfy my client. | ½ hour | 17th May 2023 | A |
|  | Research and take notes on more intense concepts of coding that will be needed to run the project. | Although the concepts are not in the IB syllabus, they needed to be learned for the project to be executed in the best way possible. | 1 month | 17th May 2023 | A |
|  | Deﬁne all success criteria and verify with both clients. Run by the supervisor/Computer Science teacher. |  | 1 day | 18th May 2023 | A |
|  | Begin the development of the User Interface of the  Project. |  |  |  | A |
|  | Complete documentation of Criterion A | Complete criterion A that supports the client’s needs and requirements. | 4 days | 19th May 2023 | A |
|  | Begin working on documentation of Criterion B. |  | 5 days | 19th May 2023 | B |
|  | Draw UML diagrams and ensure the logic of the product. |  | 1 ½ hours + |  | B |
|  | Conduct the Test plan, ﬂowcharts, and visual representation of the product. |  | 2 days |  | B |
|  | Draw the ﬁnal interface design. | Design is approved by the client and adheres to requirements. | 2 days |  | B |
|  | Complete working on documentation of Criterion B. | Have a clear idea of the desired plan and development of the product | 4 days | 22nd May 2023 | B |
|  | Contact client | Receive rough feedback and modiﬁcations on the visual representation of the project. | 5 days | 18th September 2023 | C |
|  | Begin working on the Internal Mechanisms of the project. |  | 3 months 2 weeks |  | C |
|  | Contact the client and show them a rough draft of the website. | Discuss the project's functioning and solicit additional input on its functionality and display on the page. | 1 week |  | C |
|  | Hand software to the client for Alpha testing | The client will use the website for two weeks and give feedback on any troubles or bugs that might appear. | 2 weeks |  | C |
|  | Debug code and test functionality of code | Ensure the product functions correctly before meeting with the client. | 1-2 weeks |  | C/D |
|  | Begin Documentation of Criterion C |  |  |  | C |
|  | Final Meeting with Client | Ensure the ﬁnal product reﬂects the client's wishes and needs. | 1 hour |  | D |
|  | Criterion D documentation is complete. |  | 60 minutes |  | D |
|  | Criterion E documentation is complete. | Complete final product evaluations and provide feedback on future enhancements | 4-5 hours |  | E |

## **Test Plan**

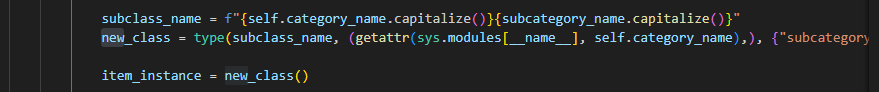
|  |  |  |
| --- | --- | --- |
| **Test Type** | **Nature of Test** | **Example Template** |
| Security Testing | Confirm that users can securely log in using their email and password. | Input valid credentials and verify successful login. |
| Student Functionality Testing | Check if students can report lost items with essential details such as email, description, and subcategory. | Register a lost item with valid details and confirm successful registration. |
| Role login Testing | Verify that users are directed to role-specific dashboards upon login for effective navigation. | Login with different roles (prefect, student) and confirm redirection to the respective dashboards. |
| Prefect Functionality Testing | Confirm that prefects can register lost items, categorize them based on subcategories, and delete items marked as found. | Prefect logs in, registers a lost item, marks it as found, and deletes it. |
| Dynamic Classes Creation Testing | Check if the system dynamically creates tables for each category and subcategory, allowing for efficient data organization. | Register lost items in different categories and subcategories, verify the creation of corresponding tables. |
| Evaluation | Evaluate if the project meets the defined success criteria. | Cross-reference the defined success criteria and confirm if each criterion is met. |
| Evaluation | Collect feedback from users on the overall functionality and user experience. | Create a feedback form or collect user opinions to gather insights for improvements. |
| Security Testing | Confirm that the application is protected from SQL injections. | Attempt SQL injection attacks and verify that the application rejects malicious attempts. |

# **Criterion C: Product development**

## **Techniques used**

**Object-Oriented Programming (OOP)**

The program carefully crafts a codebase that is based on modularity and simplicity by integrating Object-Oriented Programming (OOP) principles into its design. The ItemCategory class, a key component that dynamically spawns subclasses like TechItems, Clothes, and SchoolStuffs, is at the center of this modular design. These classes provide a modular and contained structure, with each one serving as a storehouse for the unique characteristics and behaviors intrinsic to its respective item type.



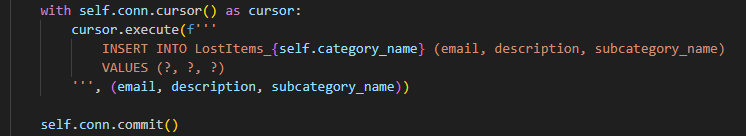
The application's use of user-defined subcategories to dynamically construct subclasses at runtime, introducing a new paradigm of flexibility and adaptation through inheritance, reveals the design's brilliance. This dynamic class generation process runs smoothly, enabling the system to easily accept new item categories and subcategories without requiring significant changes to the current codebase. Following the fundamental principles of OOP, namely encapsulation and modularity, this technique improves the maintainability and flexibility of the code while creating a solid and scalable framework that can expand with the system.

Essentially, the deliberate integration of OOP principles goes beyond simple coding rules and instead forms a guiding philosophy that shapes the fundamental structure of the application. The ensuing harmony of simplicity, adaptability, and modularity not only simplifies the existing functioning but also prepares the architecture for the future and establishes a strong basis for ongoing development and innovation.

**SQL Injection Protection**

Specifically, functions like validate\_user and register\_user heavily rely on string formatting to build SQL queries in the original code. Because string formatting incorporates user inputs directly into SQL queries, there may be a security issue because SQL injection attacks could occur. This technique could be used by malicious users to insert malicious SQL code, endangering the database's integrity. It is strongly advised to switch from string formatting to parameterized queries or prepared statements in order to mitigate this risk.

By treating user inputs as data rather than executable code, parameterized queries greatly reduce the chance of SQL injection. This entails using placeholders in SQL queries and giving user inputs as arguments in the context of the Lost and Found system. For example, using placeholders in the query and giving the password and email as parameters in the validate\_user method improves security. The required escaping and validation can subsequently be handled by the underlying database engine, in this case PyODBC, guaranteeing that user inputs are appropriately sanitized. This change significantly increases the difficulty for potential attackers to alter searches and carry out illegal actions on the database.



The program complies with secure coding requirements by using prepared statements or parameterized queries, which strengthens its defense against SQL injection issues. This major modification creates a solid basis for managing user inputs in a secure and hygienic manner throughout database interactions, strengthening the Lost and Found system's overall security posture.

**Database Management**

This program manages user credentials and missing item data effectively by integrating with a SQL Server database in a seamless manner. The system uses a dynamic table construction strategy, creating tables like UserCredentials and LostItems\_{category\_name} programmatically with the help of SQL queries. The database can adjust to changing data requirements thanks to this dynamic procedure, which also ensures scalability and versatility when managing different item categories and subcategories. The solution puts security first during user authentication by using SQL queries to verify user credentials, improving access control overall. The Lost and Found system's flexibility is increased by the dynamic table construction based on item types, which makes it easier to add new categories and subcategories without having to manually modify the database structure.

Apart from its effective data administration, the dynamic SQL interaction of the application plays a major role in enhancing the scalability and versatility of the Lost and Found system. The database structure is easily scalable to meet a variety of data needs, which is essential for a system that handles a changing variety of lost and found goods. This adaptability allows the system to grow item categories and subcategories over time, ensuring its long-term viability. The intelligent design is demonstrated by the deliberate use of SQL queries for table formation and user authentication, which is in line with best practices for developing scalable, flexible, and safe database-driven systems.

**Graphical User Interface (GUI) with Tkinter**

The program places a high priority on user accessibility and engagement thanks to an intuitive user interface created with the powerful Tkinter Python GUI toolkit. The foundation for creating aesthetically pleasing dashboards, login screens, and item registration forms is Tkinter. By utilizing Tkinter widgets, which include buttons, labels, and input fields, the application guarantees a user-friendly interface that improves the overall experience. During the item registration process, this GUI framework makes it easier to create windows with interactive input fields for the user's email address, the item description, and the subcategory. The smooth incorporation of Tkinter components offers a recognizable and visually appealing layout in addition to responsiveness, which makes for a productive and pleasurable user experience.

The program's ability to dynamically modify its interface in response to user interactions is one example of Tkinter's adaptability. Tkinter widgets facilitate the development of a unified and efficient user experience by managing login credentials and recording misplaced objects, for example. Because of the framework's interactive features, users may easily browse through a variety of functionalities, which supports a user-centric design. The application's usage of Tkinter highlights its dedication to providing a graphical user interface and is essential to the overall functionality and accessibility of the Lost and Found system.

The Lost and Found application exemplifies a comprehensive approach to problem-solving by integrating several strategies, taking into account both functional and user experience factors. The dynamic generation of tables and classes in conjunction with an intuitive graphical user interface creates a strong basis for effective lost item handling in educational settings.

# **Criterion E: Evaluation**

## **Meeting success criteria**

## User Authentication: The application implements a secure login system, allowing users (students and prefects) to access the application securely using their email and password. Utilizing Tkinter for the graphical interface, the login screens prompt users for their credentials, and SQL queries are employed to validate the provided information, fortifying the system against unauthorized access attempts.

## Efficient Item Registration: The system enables students to report lost items effortlessly by providing essential details such as email, description, and subcategory. Tkinter's intuitive widgets, including labels and input fields, facilitate a smooth registration process. Leveraging dynamic class development, the application adapts to user-defined subcategories at runtime, ensuring a comprehensive and flexible item registration mechanism.

## Role-Specific Dashboards: Upon login, users are directed to role-specific dashboards, tailoring the experience for both prefects and students. Using Tkinter, the application crafts visually appealing and interactive dashboards. The role-specific interfaces enhance user experience, providing a seamless and personalized journey within the Lost and Found system.

## Prefect's Management Capabilities: Prefects are empowered with advanced management capabilities, including the ability to register lost items, categorize them based on subcategories, and delete items marked as found. Tkinter's integration assists in creating windows with input fields, enhancing the efficiency of item management tasks. SQL injection protection ensures the integrity of data as prefects perform these critical management functions.

## Dynamic Item Categorization: The system dynamically creates tables for each category and subcategory, ensuring an organized and adaptable data structure. This dynamic categorization, implemented through SQL queries, simplifies the addition of new item categories and subcategories, contributing to the system's scalability and versatility.

## SQL Injection Protection: Prefects benefit from effective management capabilities within the application, including protection against SQL injections. The system ensures the integrity of data as prefects register, categorize, and manage lost items, meeting the criteria for a robust and secure management process.

The Lost and Found application goes above and beyond to provide prefects with advanced management capabilities while protecting against potential security threats like SQL injections. It does this by addressing essential functions like user authentication, efficient item registration, and role-specific dashboards. Dynamic class development for flexible item categorization and the smooth integration of Tkinter for an intuitive graphical user interface highlight the dedication to improving system flexibility and user experience. The program is a unique solution in the field of lost item management because of its all-encompassing approach, which guarantees the effective daily operation of the Lost and Found system while also setting the standard for scalability, versatility, and strong security measures.

## **Feedback from Client**

**Recommendation for Future Improvements**

* Enhanced User Interface: The application could benefit from an enhanced visual design and layout to provide users with a more appealing and intuitive experience. Incorporating advanced styling options and possibly introducing themes through Tkinter can significantly elevate the overall aesthetic of the Lost and Found system, contributing to increased user satisfaction.
* Notification System: Implementing a notification system would be a valuable addition to keep users informed about the status of reported items. For instance, users could receive notifications when an item is marked as found, enhancing communication and providing real-time updates on the resolution of reported incidents. Tkinter's capability to create pop-up notifications can be utilized for this purpose.
* Search and Filtering: To accommodate the growing database, consider adding search and filtering options for easier item retrieval. Introducing a search functionality within the dashboards, allowing users to filter items based on categories, subcategories, or status (found/lost), can significantly improve the efficiency of locating specific items within the system.
* Offline Mode: Enhance the application's usability by considering the implementation of an offline mode. This feature would enable users to report lost items even when not connected to the internet. The application can then synchronize the data with the server once the connection is restored, ensuring uninterrupted functionality and providing users with the flexibility to use the system in various scenarios. Integrating this capability requires careful consideration of data synchronization methods and error handling to ensure data consistency.

# References

A Detailed Evaluation of SQL Injection Attacks, Detection and Prevention Techniques. *2022 5th International Conference on Advances in Science and Technology (ICAST)*, 352-357.

<https://docs.python.org/3/tutorial/classes.html>

<https://realpython.com/python-send-email/>

<https://docs.python.org/3/library/tkinter.html>

<https://github.com/mkleehammer/pyodbc/wiki>