Feasibility Report

Open Inventory

Table of Content

Group details

> Group Number: 5
> Project Title: Inventory Management System for Computer Laboratory
> Mentor: Mr. Sachin Kahawala

Team member contribution

Index Number	Member	Contribution
170081L	CHANDRASIRI K.D.S.A.	Technical Feasibility, Social/Legal Feasibility
170097P	DASSANAYAKE D.M.G.C.	Financial Feasibility, Risk Feasibility and Mitigation
170096L	DASHINTHA E.P.D.G.	Resource and Time Feasibility, Risk Feasibility and Mitigation
170094E	DASANAYAKA D.H.M.S.N.	Considerations, Social/Legal Feasibility

1. Introduction

1.1 Overview of the Project

The proposed system is an automated inventory management system of computer labs. It should be reliable enough to be used in IoT labs and computer labs as well as to be used in any organization. Role-Based access control is to be enforced to define separate functionalities for different roles. A mobile app will be integrated into the system to make the system more user-friendly and usable to both students and the staff. The system will record inventory transaction events through the web portal and mobile app and will output inventory details, statistics, and reports. The system will communicate with lecturers through auto-generated emails to make the system more accessible for them.

This project will be an open-source, free software built for the target of letting users manage university laboratories easily. Here any potential user will have to deploy the web app and the server in his/her server and then let the users access the system. The mobile apps will allow users to connect to the specific server.

In our project, we would create the base system and the required documentation site. We would host the system as a demonstration for the end-users.

1.2 Objectives of the Project

- Ensure that the items in the inventory are tracked
- Automate the process of borrowing items from the lab.
- Handle any inventory changes including adding new items, modification of items, etc.
- Ensure that the lab management and the item borrowing is done by only permitted individuals.

1.3 The Need for the Project

Inventory management needs a lot of documentation, forms, and folders when implemented manually. There is a need for automating inventory management to ensure the safety of the items in a laboratory, make borrowing of items efficient and make monitoring of labs simple for administrators. In current systems, users are required to do all the work manually which results in a lot of wasted time. Users being unable to see the availability of lab items also makes it harder for them. So a sustainable and reliable automated system is required to make this process more efficient and easy to use.

1.4 Overview of existing Systems and Technologies

- **Inventory Management System for the Embedded lab** - Dept. of CSE
  The current system is completely manual. The records of items are in papers and borrowing of an item is done by filling a form.
- **ERPAG Inventory management software [1]** - This is a popular general inventory management software. This software records inventory items but is more suited for warehouses. This is a subscription-based online software. (starts at 49$ per month)

1.5 Scope of the Project

The basic functionalities of the system are,

1. **Dynamically configurable role-based access system.** Initially, only the administrator and student roles will be available. Administrators can create new roles with different privileges

and add accounts as staff. *Example: An administrator can create a role technical officer and grant privileges to create labs, add items, and lend items.*

2. **Registering students and staff into the system.** Registering will be done via an email sent to each student. Admin (or any user with relevant privileges) can add email addresses that are permitted to create accounts in the system.
3. **Requesting to borrow items in the lab by students.** Students can use the mobile app to view and add lab equipment(App will show availability, etc...) and request to borrow them via a supervisor/lecturer. The system will interact with supervisors/lecturers via emails to gain access on behalf of students. Students can then go to lab staff and borrow items.
4. **Lending the permitted items to the students by staff.** Through the mobile app, staff can easily scan item bar-codes and student ids to lend items.
5. **Creation of labs and assigning of staff into labs.**
6. **Maintaining the inventory of items of each lab.** Availability as well as other metrics can be easily monitored through the web application.
7. **Bar-code generation for items without serial numbers.**
8. **Marking items as damaged, unavailable and transferring items from one lab to another.**
9. **Utilize the mobile app to use for entering item bar-codes and marking items.** The mobile app will directly support selecting items through the camera. Staff web app will be connected to provide seamless integration to increase user-friendliness.

1.6 Deliverables

The main deliverables of the purposed system will be two separate **mobile applications** targeted for students and managing staff and a **web-based application** for managing staff. The web application will be mainly focusing on desktops. The mobile applications will be made to be cross-platform with simplicity in mind. Mobile apps and web application will communicate with the **central server** using a **REST API**. The users will be able to access the **documentation** on a separate website. Documentation will provide information on API details, how to deploy the system in their server, etc...

- The base system will be created with extendibility in mind. Users who want to use the system in their organizations will be able to easily change the necessary configurations and deploy them on their servers. The documentation would provide users the required direction.
- Students will not be able to sign in to the web application. All the student functionalities will be implemented in the mobile app.
- Staff members will be able to use both the mobile app and the web app. The mobile app will mostly act as a helping tool for them.

2. Feasibility Study

2.1 Financial Feasibility

As mentioned in the overview, this project will be an open-source free software where the hosting and deploying the system has to be taken care of by the lab administration. However, for demonstration, we will host and deploy the web server in a minimum viable manner.

Expenses

Web-app and database hosting

Since the proposed system is a centralized client-server application, web app hosting is a major cost. Here for the demo, we plan to use Heroku to host the web app. To fulfill this purpose, Heroku free plan will be enough. This will also allow us to deploy the web app and Postgres database **without any cost** at the expense of limited performance and response time. The database will have a restriction on the number of rows in the free tier, but this constraint will not matter much since this will be a demo website.

Documentation pages will be static. Thus, they will be served using Github Pages. This will not result in any cost.

Multimedia hosting costs

We will use a separate CDN service(Cloudinary) to deliver images in the demo. We will use the free tier in this service. Even though the free tier allows us to use only 25 credits* monthly, because of the limited number of users and low usage of images, the required bandwidth will be very low. So we will not face any issues when using the **free plan**.

*1 cloudinary credit = 1GB managed storage or 1GB net viewing bandwidth

When a user decides to use this system in their laboratory, he/she can change the system configurations to use their CDN service easily. Since the scope of this system does not include many multimedia uses, the free tier would be enough for any user of this software.

Play Store/App Store deployment

> At the initial stage we will not target iOS deployment since the cost of creating an app store account is very high.

Store	Cost
Play store(Android App)	25$ registration fee
App store(iOS App)	99$ annually

We will initially publish the Android application to the play store. This app will be similar to that of the Moodle[2] app. The users(students/staff) can enter the server address and use the mobile app in their system.

This will be a cost on our side, but since it would be a one-time payment, it wouldn't be a major issue.

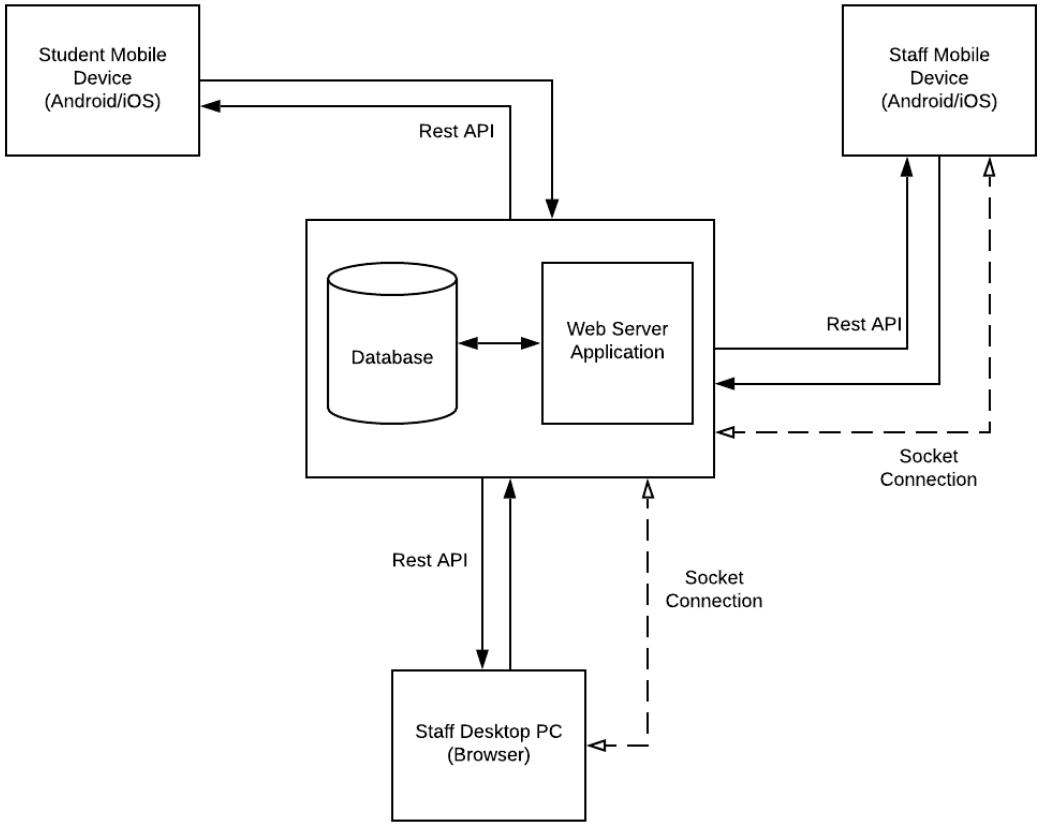
Income sources

The system will follow the freeware software standards. **No cost will be charged from the potential customers**.

Apart from the associated cost, there will be many benefits for the customers. Especially the extra effort that is associated with completing forms and keep records of equipment that is borrowed from the lab will be significantly reduced.

However, after initial deployment, we will not face any cost because the customers would have to host the application in their environments.

2.2 Technical Feasibility



Technology stack

This project is a client-server application based on multiple types of clients. Base technologies that this system will use are,

- **NodeJS** - Back-end
- **React** - Front-end web application
- **Flutter** - Front-end mobile application
- **Postgres** - Default Database system*

*Even though Postgres is supposed to be the default database, the system will be made to be database agnostic as possible.

Other important technologies

Back-end will use **JSON Web Tokens**[3] as the primary method of authentication. Also, the system will have a **role-based authentication**[4] system implemented from ground-up to facilitate the requirements. For JWT verification, jsonwebtoken package will be used.

Staff mobile app will have **barcode scanning** capabilities for the app to be used as a barcode reader. We plan to use barcode scan third-party library for this task.

To connect the mobile app and the web app in real-time we will use **web-socket connections**[5]. For this feature, we plan to use socket.io implementations on Node, React and Flutter. Here we will use a unique room for every user to share data between the same user's devices. Using this approach, we can let the user scan a barcode in the mobile device and select items in the web app seamlessly.

Automated Emails will be sent through node mailer. Other automated tasks such as automatically checking for due date expired items, etc... will be done via **CRON jobs**[6] using bull package.

To **host user uploaded image content**(eg: item images/lab images) we will use cloudinary CDN service. Even though Cloudinary does not have a react integration SDK, we can use the existing JavaScript SDK to upload content into the network.[7]

To make the system easily usable, we will make the system as **database-agnostic** as possible using the sequelize ORM[8] allowing the users to use any database of dialects: PostgreSQL, MySQL, MariaDB, SQLite, and MSSQL.

Conclusion

According to the above resources and initial analysis, this project is technically feasible.

2.3 Resource and Time Feasibility

Resource feasibility

Resources that are required for the product include the following. Since this project does not require novel equipment or technologies, this project does not have issues in resource feasibility.

- Programming device (Laptop)
  The development crew already possesses devices with the required level of processing power. Mobile devices required for testing are also available with the development crew.
- Hosting space (freely available)
  Hosting is to be done by the separate institutes which use this software. However, we would require a hosting server to host the demo project. Required hosting spaces for these tasks were noted under 2.2.
- Programming tools (freely available)
  Some of the programming tools used for system development are noted below. Every tool is freely available for commercial use.
  - Visual Studio Code - IDE for Web and Mobile development
  - pgAdmin - Admin panel for Postgres database
  - Typora - Generation of markdown documentation
  - Postman - Debugging server API
- Programming individuals
  The project team composes of 4 individuals with adequate potential.

So it is clear that the project has the required resource feasibility.

Time feasibility

The project duration is 14 weeks. The project work will be distributed across the 4 members available accordingly.
For more details please refer the project schedule document.

2.4 Risk Feasibility and Mitigation

To reduce system issues related to bugs, we will extensively test the system. Apart from that, the following measures will be taken to mitigate other risks.

System fail

In the case of system failure, the manual system should be continued. In that case, users should be able to continue the manual system as well as the automated system at the same time. For this specific requirement, our system will contain an additional item state. When an item is tracked under the manual system, staff can simply put that item under that specific state when the system restarts. Then that item will show up as unavailable to the students and the staff can reset the state when the item is reacquired.

Database Backups

Database backups will occur weekly(system administrators can change the frequency) to mitigate database failures.

Security issues

The system will keep a log of every important action done by every user to make sure that tracking a user or an item is possible in case of some item going missing.

To authenticate users, the system will use an email-based process, where users have to confirm account creation through an email sent to the email account given by the administrator. This will ensure that unauthenticated users will not be able to access the system. Apart from that, standard hashing and encryption methods will be used when storing user passwords and generating JSON web tokens.

2.5 Social/Legal Feasibility

Social Feasibility

The main users of the system are the students and the laboratory staff. A manual system would require both parties to go through a lengthy process for use-cases such as borrowing an item or adding an item to the inventory.
Most of these processes will be made much easy and reliable through the system. So the stakeholders will prefer this system over legacy manual systems. Yet constant user input during the development process and prior acknowledgment of users will ensure user cooperation.
The technical know-how of some users might be not adequate to use the system. In such a case users might have to be trained.

Legal Feasibility

Legal implications of the system

The system uses freely available development tools and is provided as an open-source system.

Although the system should deal with the private information of the institute, the developing community has no access to the data. The hosting and the maintenance is carried out by the relevant educational institute. So there will not be legal concerns regarding consumer data protection.

The system will be distributed as free and open-source software with **MIT** licensing[9] to ensure the intellectual property rights preservation and the ability for using for commercial license.

The work would be provided "as is". A user may not hold the authors liable.

Third-party Licensing

The core technologies used for the development of the system: React, Flutter, and Node are open-sourced under the following licenses.

Framework	License
NodeJs	MIT License
Flutter	BSD 3-Clause "New" or "Revised" License
React	MIT License

Also, every library mentioned in this document uses the MIT license. So they might not require any legal procedures for the use.

3. Considerations

- **Usability and Ease of use** - Users should be able to easily adapt and use the system. Even technologically challenged users must be able to easily understand and use the system. UI/UX should be simple and intuitive.
- **Maintainability & Flexibility** - System should be able to maintain for a long period and the system should be able to adapt to any new design changes in the base system.
- **Security of the lab items** - Security should be high in the system since the entities involved are costly lab equipment.
- **Reliability** - The system should not fail since that would disrupt a whole section of the university/organization that is using this system. Also, there should be a fail-safe mechanism in case the online system fails. (eg: the manual system which can be later synced with the online system)
- **Clear and concise documentation** - Since this system is supposed to be extended by system users, clear documentation should be present.

Since this is not a system with neither a huge number of users nor a time-critical application, performance is not a huge consideration. **However the system should be able to handle at least 250 concurrent users and the response time should not exceed 5 seconds.** These performance criteria will be taken into account when designing the system.

4. References

[1] ERPAG. (2020). Inventory management software - ERPAG. [online] Available at: https://www.erpag.com/inventory-management-software [Accessed 2 Mar. 2020].

[2] Docs.moodle.org. 2020. About Moodle - Moodledocs. [online] Available at: https://docs.moodle.org/38/en/About_Moodle [Accessed 10 March 2020].

[3] Jwt.io. 2020. JWT.IO - JSON Web Tokens Introduction. [online] Available at: https://jwt.io/introduction/ [Accessed 10 March 2020].

[4] En.wikipedia.org. 2020. Role-Based Access Control. [online] Available at: https://en.wikipedia.org/wiki/Role-based_access_control [Accessed 10 March 2020].

[5] Krill, P., 2020. Socket.IO Javascript Framework Ready For Real-Time Apps. [online] InfoWorld. Available at: https://www.infoworld.com/article/2607757/socket-io-javascript-framework-ready-for-real-time-apps.html [Accessed 10 March 2020].

[6] OSTechNix. 2020. A Beginners Guide To Cron Jobs - Ostechnix. [online] Available at: https://www.ostechnix.com/a-beginners-guide-to-cron-jobs/ [Accessed 10 March 2020].

[7] Cloudinary.com. 2020. React Image And Video Upload | Cloudinary. [online] Available at: https://cloudinary.com/documentation/react_image_and_video_upload [Accessed 10 March 2020].

[8] En.wikipedia.org. 2020. Object-Relational Mapping. [online] Available at: https://en.wikipedia.org/wiki/Object-relational_mapping [Accessed 10 March 2020].

[9] Opensource.org. 2020. The MIT License | Open Source Initiative. [online] Available at: https://opensource.org/licenses/MIT [Accessed 10 March 2020].