King Fahd University of Petroleum and Minerals
Computer Engineering Department

Senior Design (COE485)
Semester 141

Librarian

Final Report

Team Member:

- Wael Elagi 200916510
- Mahdi Al Mayuf 200962830
- Abdulrahman Alkhiary 200947350

Advisor:

Dr. Ahmad Khayyat 20/12/2014

Table of Content

List of Tables

List of figures

1. Introduction

This project was proposed by the Computer Engineering Department (COE Dept.) to improve the department resources management. The COE Dept. has a lot of resources, but the scope of the project is to deal with the electronic parts and devices. Tracking these parts is done manually in the current procedure which makes it hard and sometimes impossible to track and account for parts. This project is about developing an information system that keeps records for all the parts in the COE Dept., and track them using a data entry mechanism to be used by multiple types of users.

2. Problem statement

The existing procedure has many drawbacks, it consist of paper logs and Excel sheets, which makes it hard to keep track of the parts going through a huge amount of paper and sheets. In addition, the forms are not unified, there are parts that are not tagged to be distinguished from other parts see Figure 1.

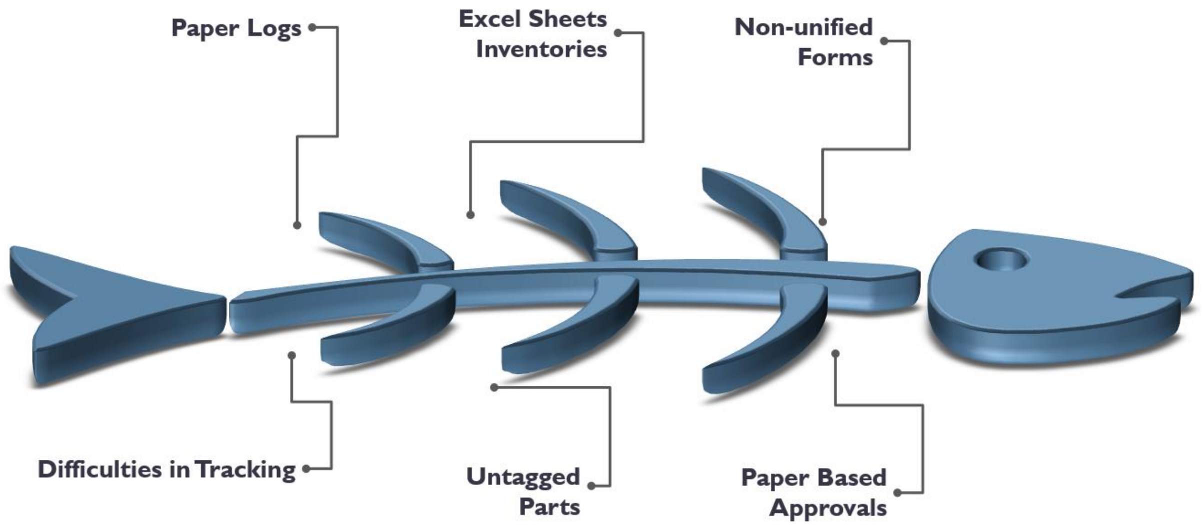


Figure 1: Drawbacks Fishbone Analysis

The project is a system for tracking electronic parts of the COE department, and managing part borrowing through approving process. It should allow its users to check the availability of a given part and to borrow it. A part is available if the department has it and it is not checked out (borrowed).

Project impact

1. Saving time, efforts and recourses.
2. Accessibility.
3. Automation of the request/approve process (Less paper work).

3. Requirements

- Add new parts and their quantities. Similar but non-identical parts should be distinguishable. Include part photos.
- Flexible categorization of parts
- Allow students to request parts, the request should be approved by faculty.
- Allow faculties to request parts.
- All requests and returns need to go through the staff for scanning.
- Only the Admin can change the privileges for each user.
- Keep tracking for request date, borrowed date, due date and returned date.
- Track individual parts to show whether a part is available or borrowed.
- Filter parts by category.
- Configurable privileges.
- Deployment of the system in a way accessible to all intended users.

4. Specification

- The system is built as a web application system to be accessible for all the users.
- A database that keep records of parts and their quantities, users, requests and approvals
- Catalog to view the parts availability and status.
- Generating a unique QR code for each part.
- Using a scan gun to read the QR code and serial code.
- Photo uploading to upload a photo for each part.
- The system will provide a different views for different user depends on their privileges.
- Deploying the system on a webserver to be accessible by all users.

5. System design

System Architecture

The system consists of two main components, Server and Client. The Client is basically a web Browser, and the server has three primary components which are, web server to serve HTTP requests, a web application server to serve the business logic, and a database. The web application server includes three components:

1. **User management:** user types, privileges.
2. **Request management**: request generating and approving
3. **Part management**: Part information, status and accountability.

See Figure 2:

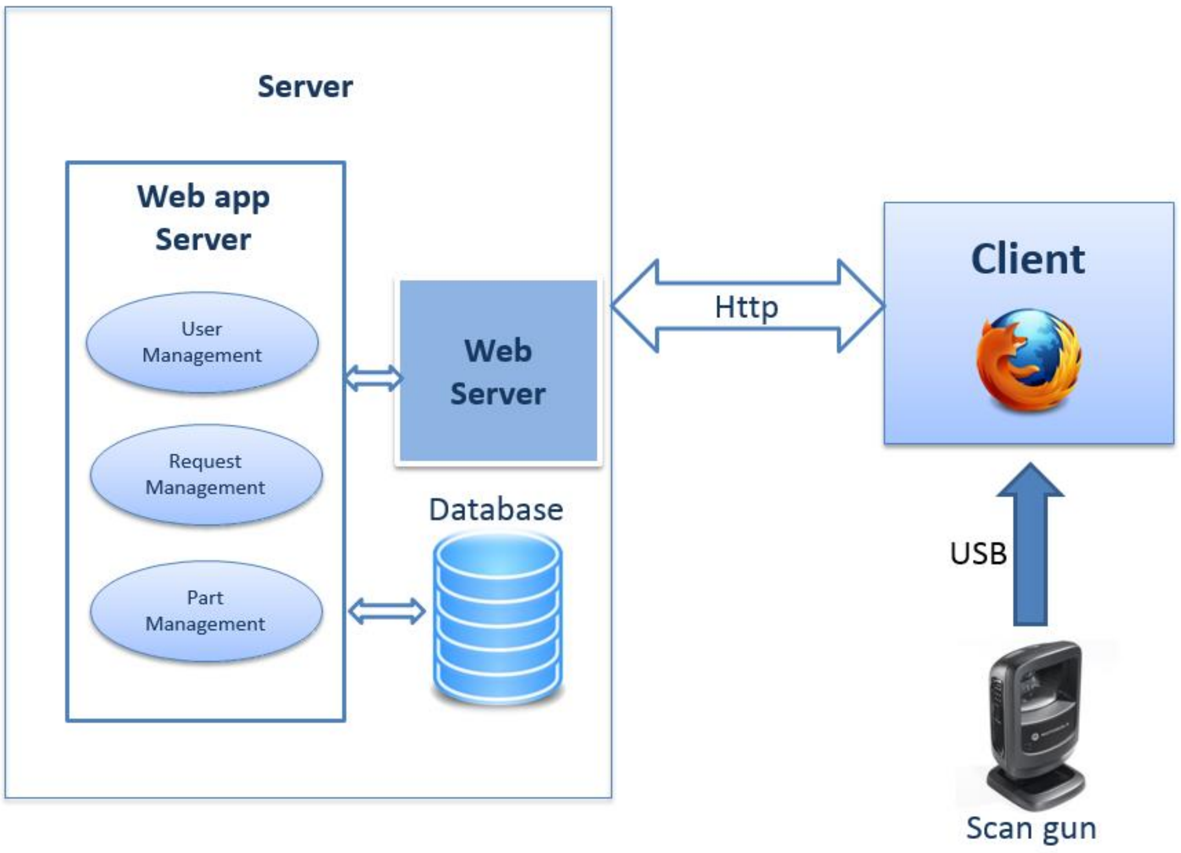


Figure 2: System Architecture

Component Design and Implementation

Components

The system has three types of components, readymade, semicustom and custom. The readymade components are hardware and software components that are configured and plugged to the system. The semicustom components consists of software components that are available but massively customized to fit the system. Finally, the custom components which are components that were built from scratch see Table 1:

Readymade

Table 1: Off-shelf Components

Hardware components	software components
• **Server (workstation)**	• Web Application server
• **Scan gun**	• Web server
	• Database server
	• Client (web browser)

Semicustom

The user management component is a semicustom component that involves authorization and authentication and other functionalities, there were available libraries, but putting them together and implementing other user management functionalities required heavy customizations and modifications.

Custom

The custom components are all software, so they need to be implemented to meet the system custom specification. These components are:

- Database
- Part Management
- Request Management

Design

The web application is going to be implemented using Asp.net MVC 5 Framework the reasons behind this choice and further elaboration on frameworks are going to be explored later in the report. There is also a hardware component which is a scan gun that is used to scan QR codes. This component is connected in the client side (web browser).

Database design

The database schema was initially designed by trying to find the different main entities of the system and their relations, but of course, when the team as developing the application the scheme evolved and changed multiple times in order for the system to work as expected, the initial ER model is shown in Figure 3:

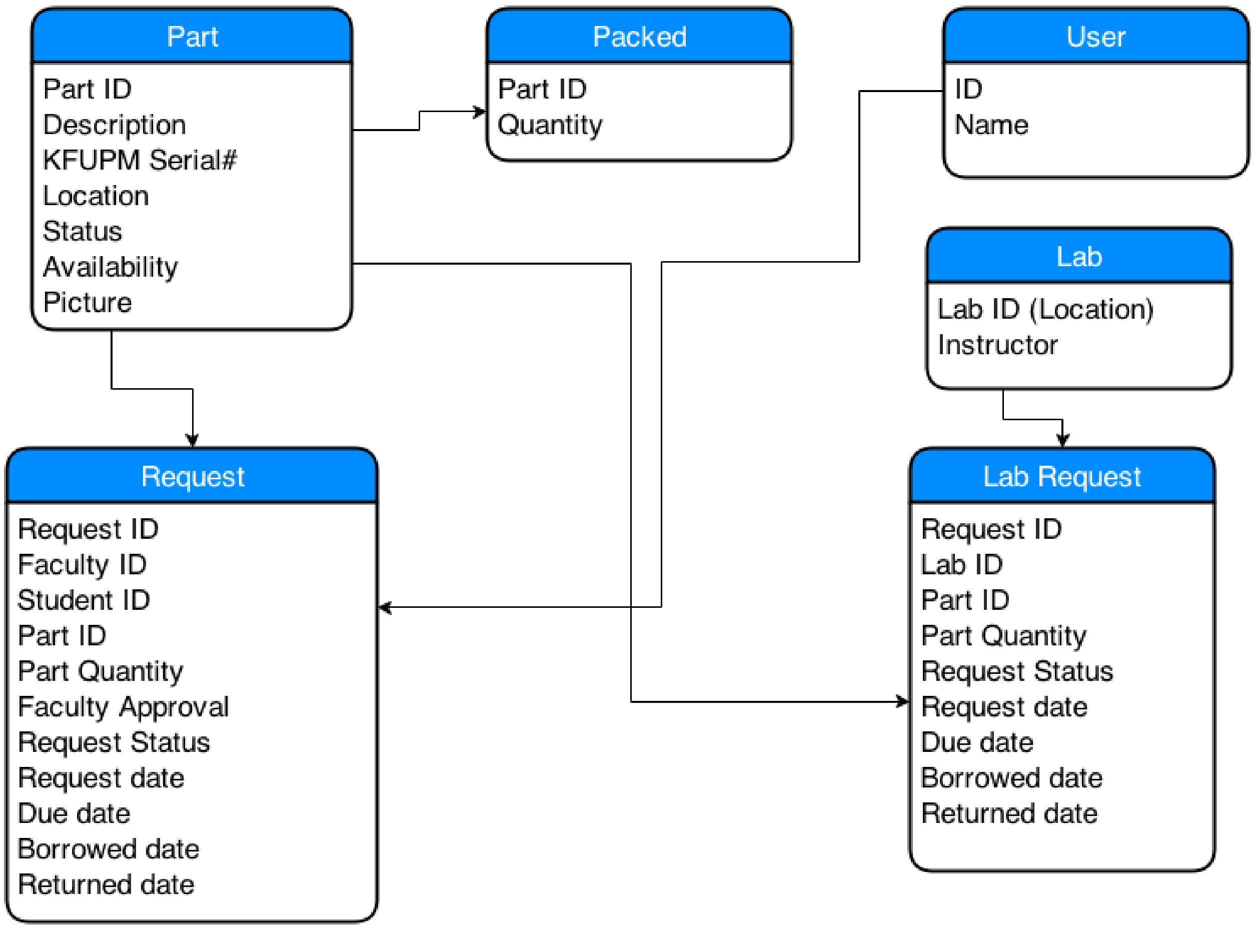


Figure 3: Database ERD

As mentioned before the previous schema has changed multiple times in order to add new entities or change some of the entities' fields, Figure 4 and Figure 5 will show the final schema (Entity- Relation Diagram). The database is logically divided into three parts, one associated with part and request management, another associated with the user management, and the last is used to some data needed by the system.

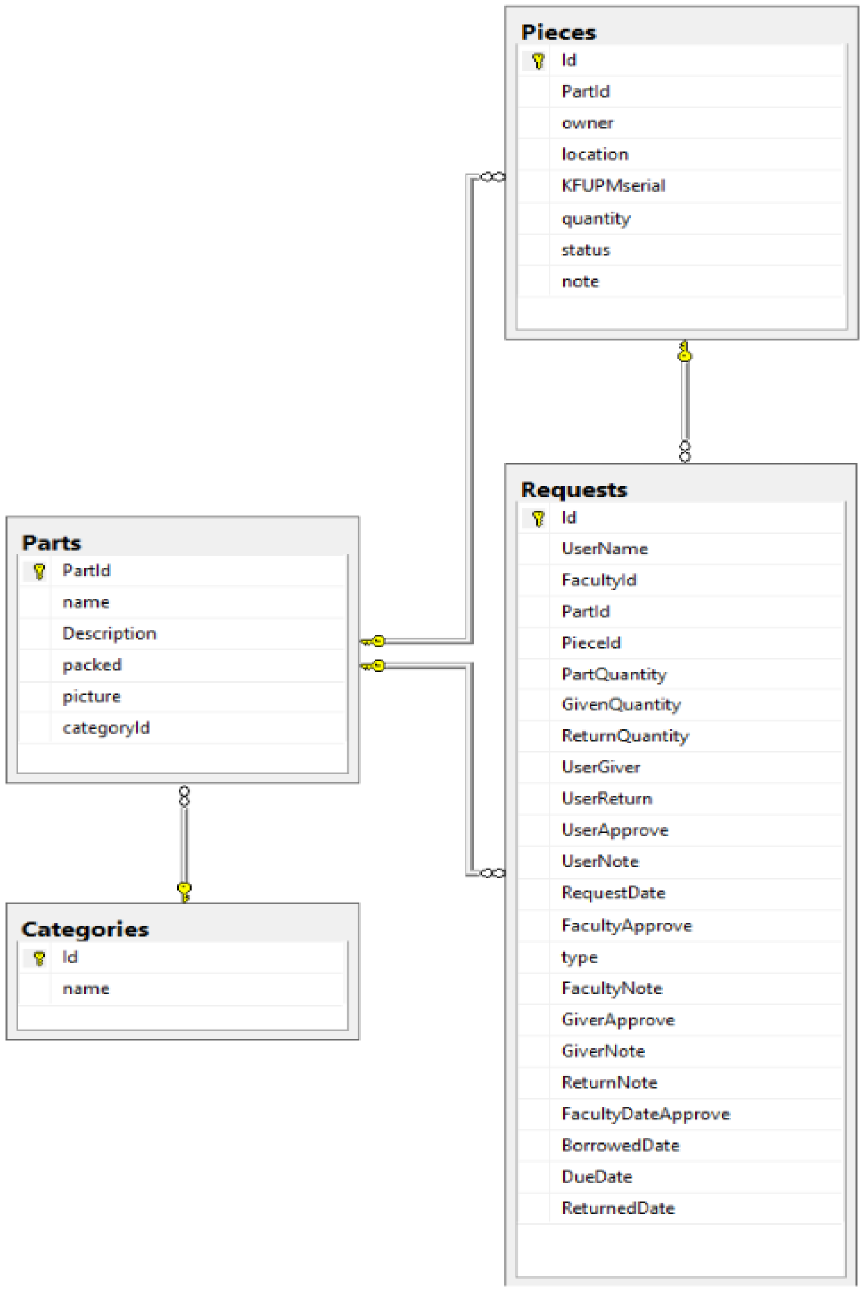


Figure 4: Part Management DB schema

The database schema associated with the part and request management consists of four entities:

- **Part:** this entity hold information about the part, it like a templet for the part information.
- **Piece:** this entity represent the actual part, since the department usually have multiple identical parts or equipment, so the part entity holds the information that are the same in all the parts while the piece hold the unique information of each part, the part entity is like the templet while the piece is the real instant.
- **Request:** this entity holds information about the request.
- **Category:** holds information about categories.

The following part of the schema is the part associated with user management:

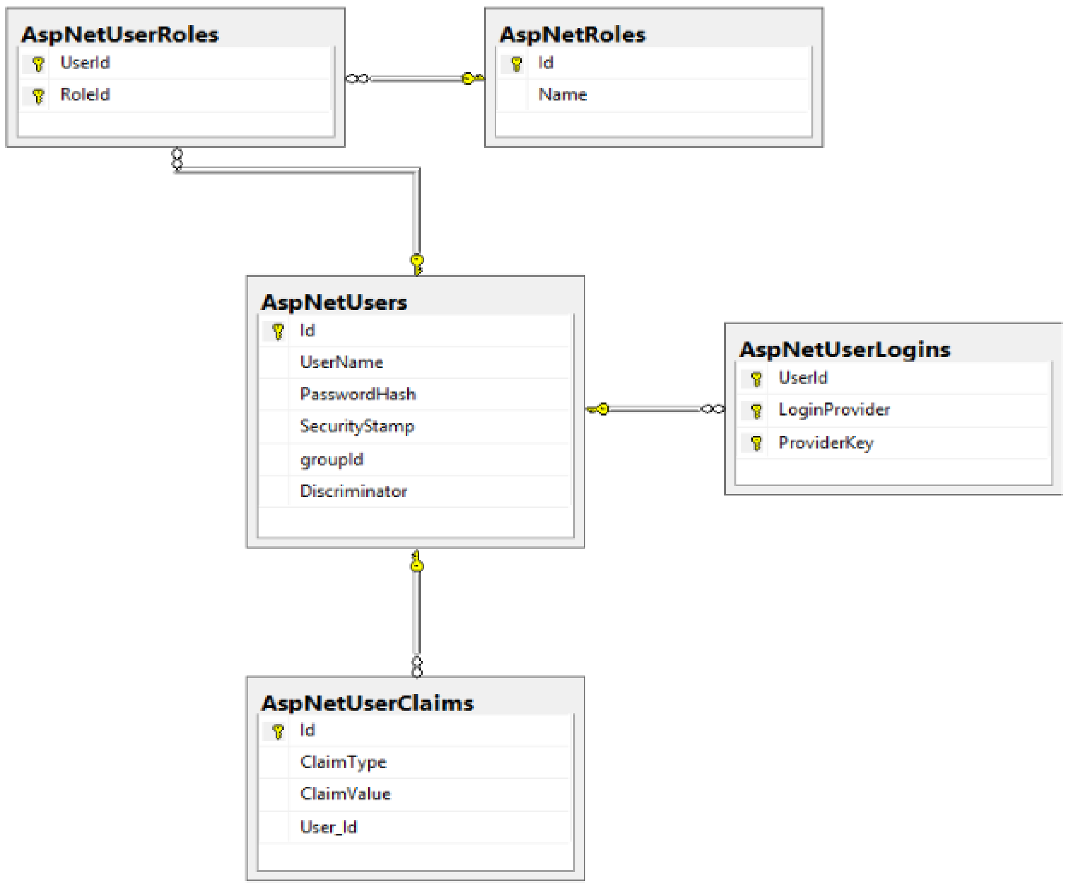


Figure 5: User Management Schema

This part of the schema holds information about the users and their logins as well as roles. Roles in this context means authorization (what is the user allowed to do in the system).

Other Software Components

The other software components are going to be designed as followed:

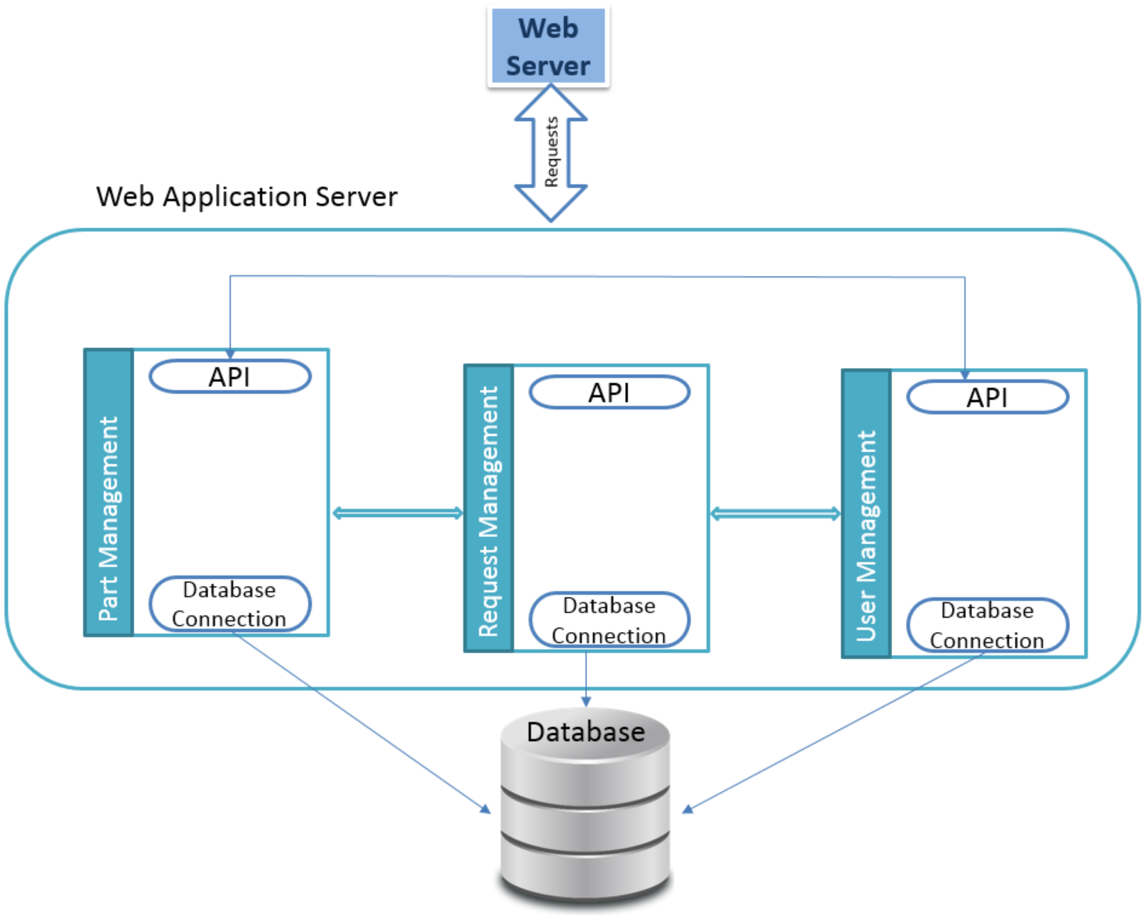


Figure 6: Web Application Design

There are three main components which are User Management, Request Management, and Part Management. Each one of these components has an API that specifies their operations and database connection to store and retrieve data.

Web application design patterns

The web application is going to be designed using the MVC design pattern the reason behind this choice is going to be explored in the design alternatives section later in this report. According to Microsoft Developer Network the MVC design pattern divides the application into three parts: Model, View and controller (MVC), see Figure 7 (Chrome Developers, 2014):

- **Model:** deals with data, and respond to query requests and state information requests.
- **View:** user interfaces (Views).
- **Controller:** the connection between the previous two, it respond to the user actions by retrieving the data models and the required views then render them.

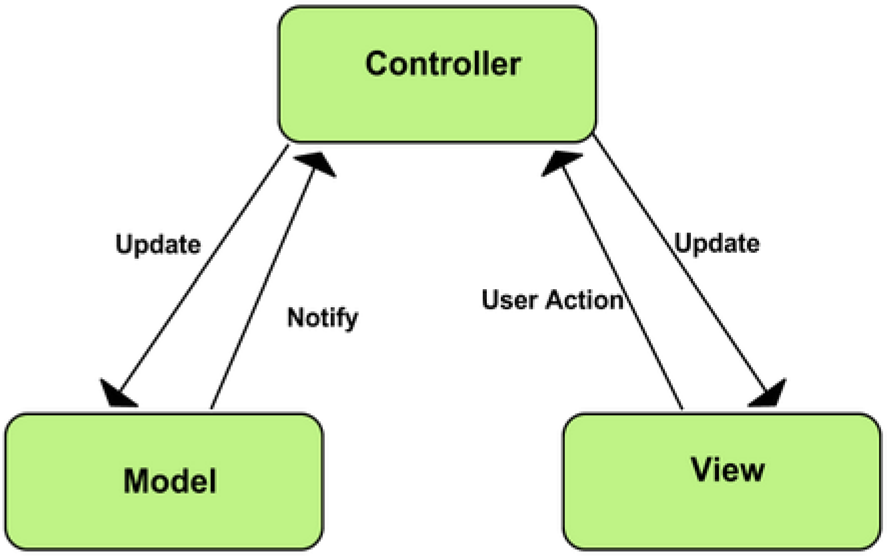


Figure 7: MVC Patterns

Implementation

Controllers

The application is going to be implemented using Asp.net MVC 5 framework. As stated before the application has three main:

- **User Management:** consists of two controllers
  - **AppUser**: that manages adding, editing and the activating of users.
  - **Account**: that manages logins, logouts.

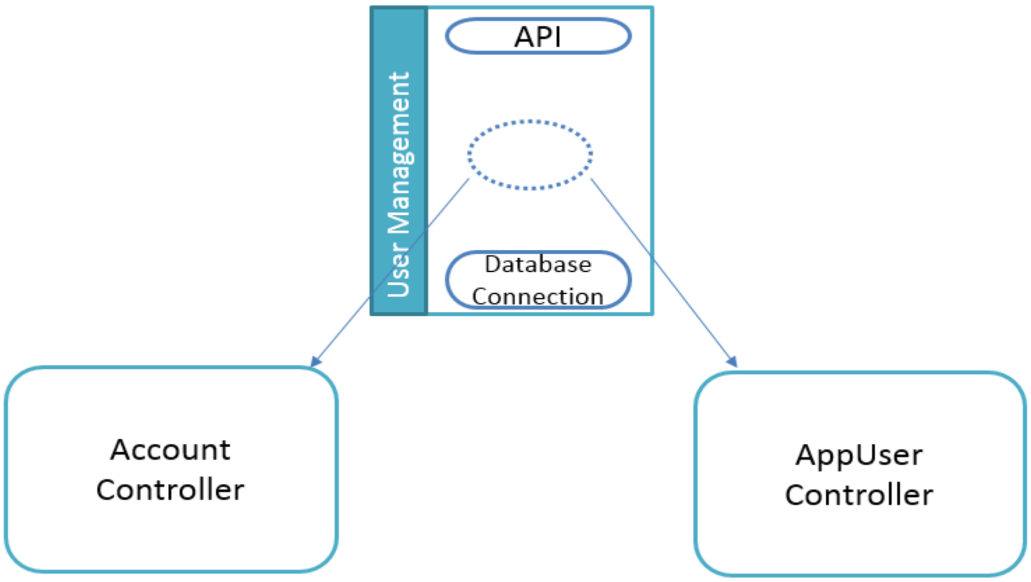


Figure 8: User Management Component

- **Part Management**: consists of two controllers
- **Part**: that manages adding, editing, accounting and tags generating of parts and other part managements.
- **Category**: adding, editing and managing categories.

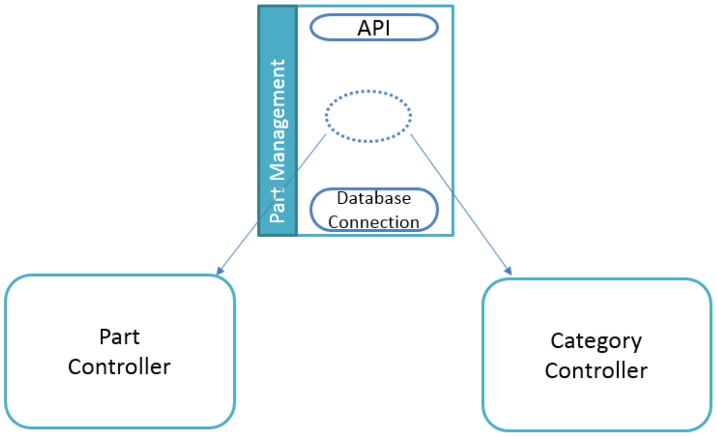


Figure 9: Part Managements Component

- **Request Management:** consists of two controllers
    - **Search**: manage parts search.
    - **Request**: manage the requests and approvals.

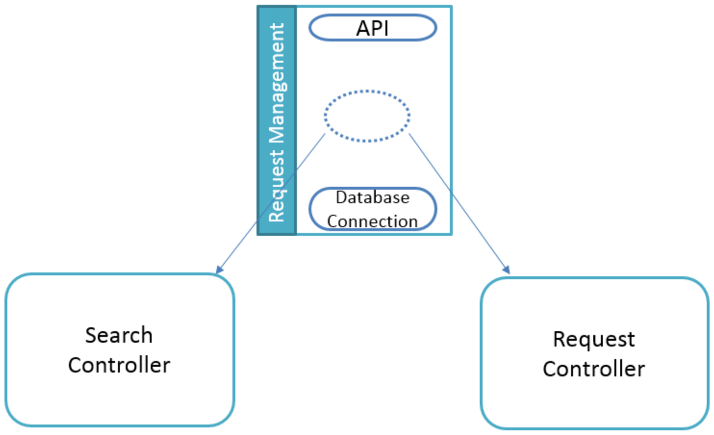


Figure 10: Request management Component

Models

Accessing, creating and manipulating the data will be performed using object relation mapping technique by using the Entity Framework. The entities in the application are represented by models (classes), and these models are used to do data operations.

System Integration

For the main components which are the client and the server, they interact using HTTP requests. Software components are interfaced using their API, and they interact through the database. Finally, hardware components like the scan gun, it interact with client side commuter through USB and give a keyboard input.

Design Decisions

There are two options to implement the system. The options are web application and Desktop application. The following table shows the tradeoffs between the two options.

Table 2: Platforms Tradeoffs

Options / Criteria	web application	Desktop application
Accessibility	Anywhere	local device
Development	One system(multiple views)	multiple system
Installation	One side	Many sides
Component integration	One environment	Many environment
System Update	Server side	All workstations
OS independence	No	yes
Server Load	High	Medium

The following table compare the options depends on the criteria's weights. The comparison scale used from 10 – 0 (10 Max, 0 Min)

Table 3: Scaled Comparison

Options / Criteria	Weight	web application	Desktop application
Accessibility	0.3	10	7
Development	0.2	7	5
Installation	0.15	8	5
Component integration	0.05	7	6
System Update	0.1	8	5
OS independence	0.15	9	1
Server Load	0.05	4	**7**

The following table shows the final result.

Table 4: Comparison Result

Criteria \ Options	web application	Desktop application
Accessibility	3	2.1
Development	1.4	1
Installation	1.2	0.75
Component integration	0.35	0.3
System Update	0.8	0.5
OS independence	1.35	0.15
Server Load	0.2	0.35
Result	8.3	5.15

From the previous tables, the web application has more features than desktop application. As a result, the project will be implemented as a web application.

RDBMS VS ORDM

The Relational Database Management System (RDBMS) is a database that is based on relational model that stores data in the form of related tables. On the other hand, the Object Relational Database Mapping (ORDM) is a programing technique that utilized through an Application Programming Interface (API).

ORDM is easier in development side than RDBMS which allows developers to convert data from rich data types used in object oriented programming languages to lower level relational database types. Also, ORDM has high efficient than RDMBS in the small/medium applications that aren't accessing the database frequently. The database schema will be implemented using ORDM. So rather than accenting the database through coded SQL statements, the team decided to use ORM to gain more flexibility and database abstraction using Entity Framework.

Frameworks

Web application frameworks (like other software frameworks) are used to eliminate the overhead of the common activities associated with every web application like templating, database access, and session management. However, deciding what framework to use, the application design pattern should be determined. There are multiple design patters such as Model-View-Controller (MVC), web forms and other patterns.

The MVC pattern holds many advantages such as:

- **Design clarity**: the code is easy to understand.
- **Modularity:** component can easily be plugged in and out.
- **Growth:** the application can grow by adding component and some of the old components can still be used.
- **Efficiency:** when the user applies an action on the system, the action gets immediately invoked unlike web forms where the complete behind code life cycle gets invoked.

In our the application we seek abstraction, since the team is going to develop the system in a Components Based Development (CBD) fashion where each component of the system is developed apart from the other components and that is found in the **"Modularity",** also the system is going to be maintained by people other than the development team so "**Design clarity**" is essential, finally we are also seeking scalability (**Growth**). Because of the previous, MVC is an ideal design pattern for this application.

After deciding the design pattern, now comes the choice of the frameworks. There are multiple frameworks for MVC, but the ones that stick out are:

- Play Framework (Java).
- Django Framework (Python).
- Asp.NET MVC.

Since the projects needed to be developed in a short time period, Django was excluded because the team doesn't have a background on Python programming, and using Python would have lengthened the learning curve. The team initial plan was to develop the application using Play Framework but due to the poor documentation, and the Framework had a new release at the time of the development with major changes, the team decided to use Asp.net , since it had good documentation, it is easy to use and it serves the purpose.

QR Code

In order to differentiate between the different parts the application generates a tag to each part. There are multiple choices of tags format, but since the university uses bar codes, the team decided to use QR codes since it can be easily differentiated from bar codes.

Figure 11: QR code

Of course in the client side the staff will need a QR code reader to read the tags. There are multiple types of QR code reader, but what is important is how the reader communicates with the PC. In that matters there are multiple communications methods used by readers, but the most used ones are:

- **RS232 Serial**: where the reader communicates with the PC over the serial port (Carolina Barcode Inc., 2014).
- **USB:** where the scanner is plugged using the USB port, and recognized as a keyboard (Carolina Barcode Inc., 2014).

The team decided to the USB (Keyboard input) since it is easy use, replace and does not require any drivers. This also remove the overhead of reading the serial input.

Figure 12: QR Scanner

Other Design Decisions

The initial design of the system had a cam attached to staff client side in order to take pictures for the parts, the team decided to replace that by picture uploading. This way add more flexibility since any device with camera can be used, and the system will not be tied to a specific type of camera.

Design Evolution

Throughout developing the application, the team needed to change the design sometimes in order to enhance the design, add functionalities or simply the design doesn't behave in the required way. Changes occurred on multiple levels, system level, application level and component level.

System Level

The Changes on the system were small, one of which is removing the camera form the design on order to avoid another battle neck beside the scanner (checkout need to go through the staffs).

Application Level

The initial design had one controller for each component, but the application ended up by using two controllers per each monument component in order add more "**Modularity"** to the application and more abstraction.

Components Level

Database is clear example on component evolution. As mentioned before the database changed multiple times either by adding new entities or changing the entity's fields in order to add some needed information.

6. System Behavior

The System Behavior consists of models that describe aspects of system which characterized as either emphasizing static or dynamic information. These models show the behavior of the system, the interaction between users and system components, communication of the system components and sequence order of each function.

Static view describes the structure of business objects, attributes, operation and relation. It includes Class diagrams and Use Case diagrams. On the other hand, Dynamic view describes the behavior of the system over time using set of states that occur in a specific order. It includes Sequence diagrams and Activity diagrams.

Static View

Class Diagram

Class Diagram provides an overview of the system and describes the system structure. Also, it shows the attributes and methods of the system classes.

The system **Models** has two class diagrams, Part and Request Management Class Diagram and User Management Class Diagram. Both these diagrams show the classes attributes and methods see **Error! Reference source not found.** and **Error! Reference source not found.**.

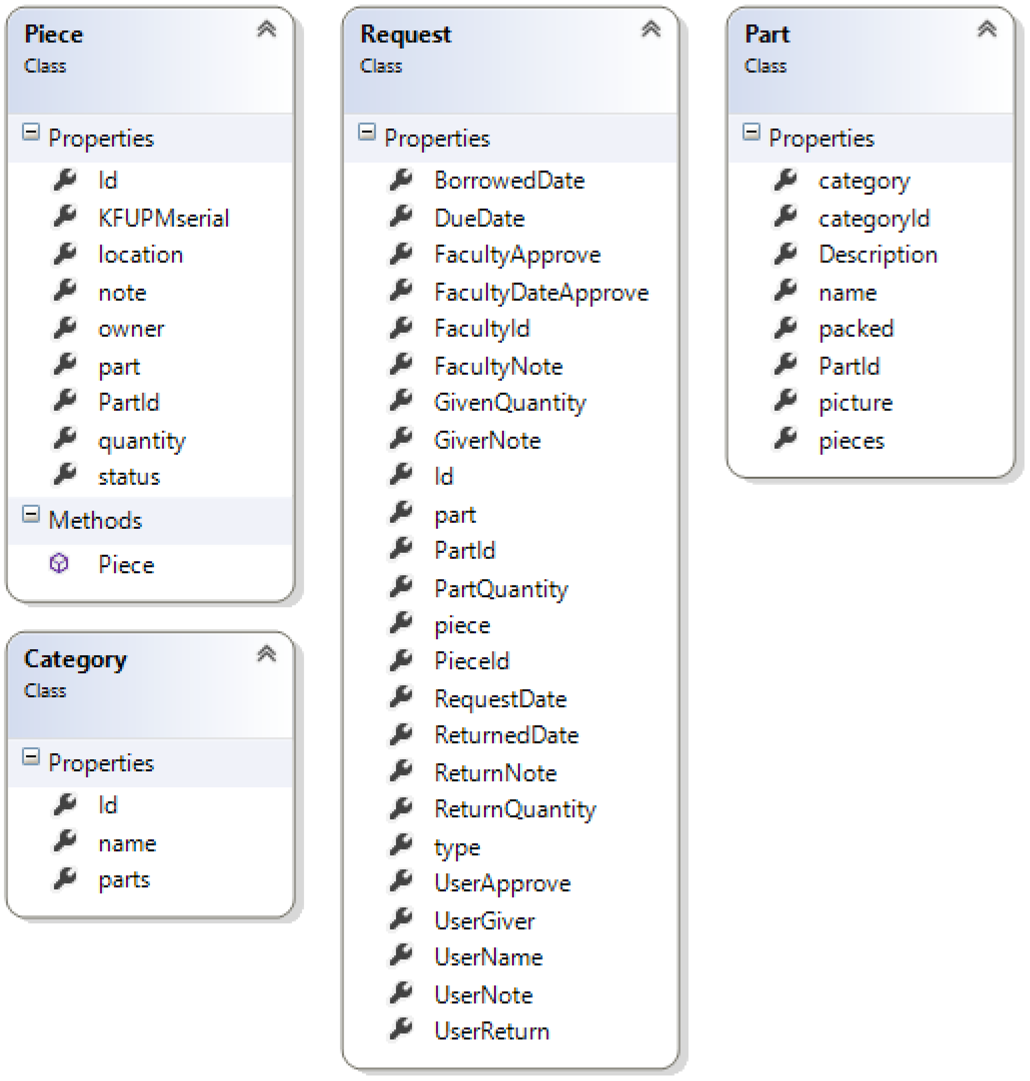


Figure 13 Part & Request class diagram

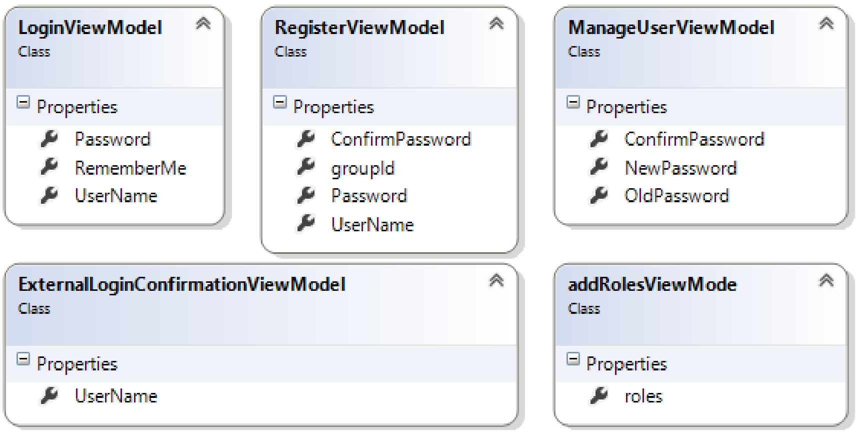


Figure 14 User Management Class Diagram

The **Controllers Class Diagram** shows the controllers methods.

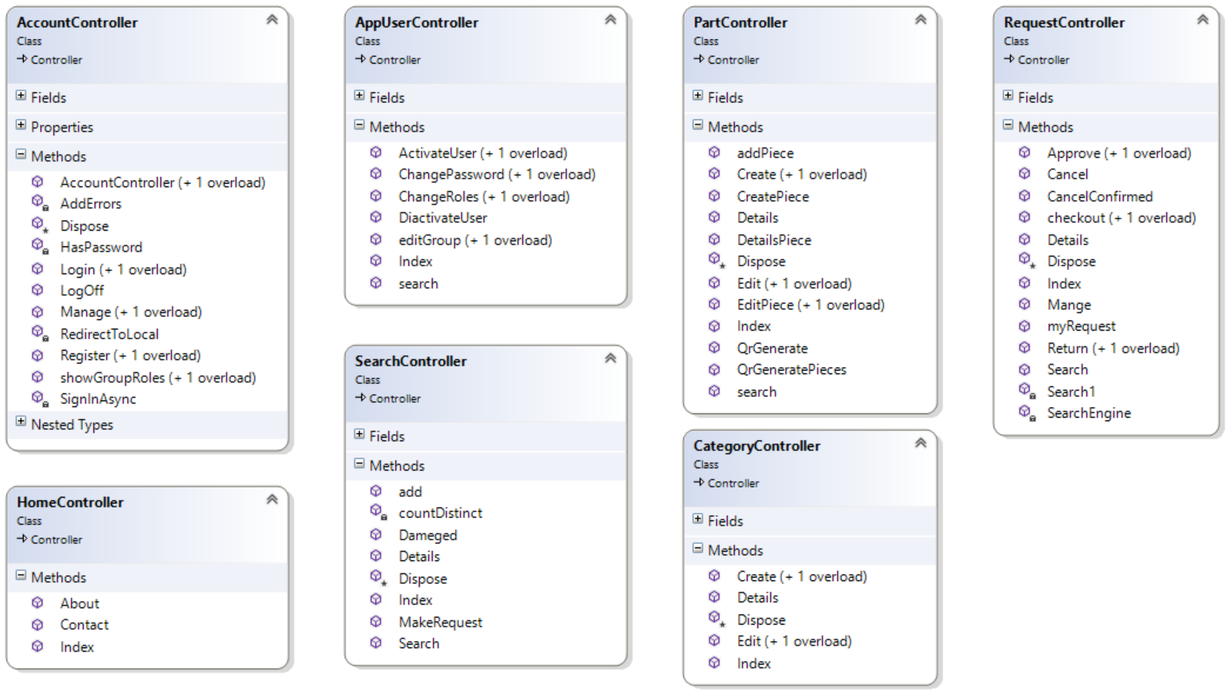


Figure 15 Controllers class diagram

Use Case Diagram

The Use Case Diagram is used to represent the user interaction with the system elements. In the Librarian system, there are four types of actors which are: Admin, Faculty, Staff and Student. The diagram shows what the users can do.

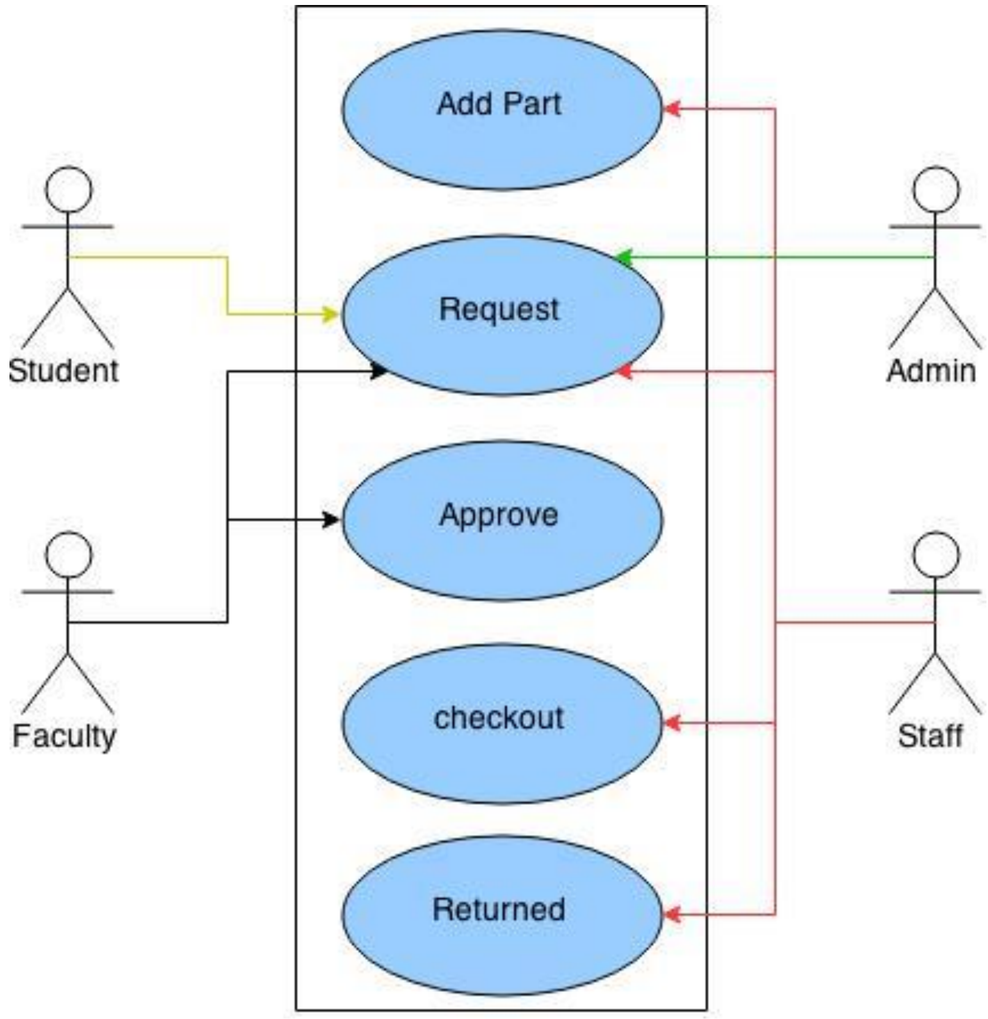


Figure 16 User Case diagram

Dynamic View

Sequence Diagram

Sequence Diagram shows the interaction between objects based on their sequence order. Each user has different sequence that depends of his role.

The **Admin** diagram shows the object orders to do his functions. First he needs to login in the system, then he can access the User Management.

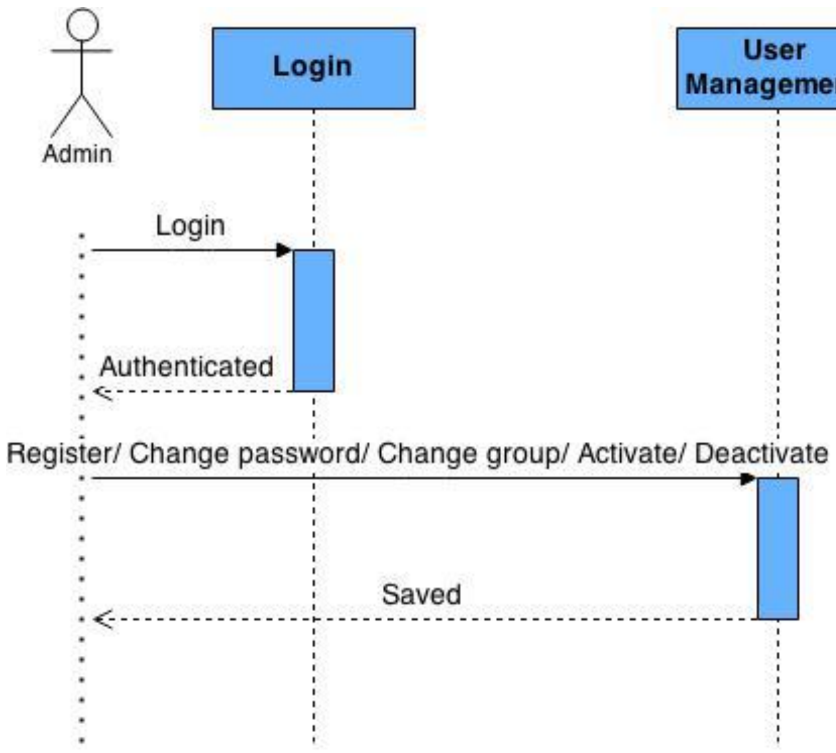

Admin
Login
User Management
Login
Authenticated
Register/ Change password/ Change group/ Activate/ Deactivate
Saved

Figure 17 Admin sequence diagram

The **Faculty** has two sequences diagram. The first diagram Figure 18 shows the sequence order for faculty to request parts. Second diagram Figure 19 shows the sequence order for approving the requested parts.

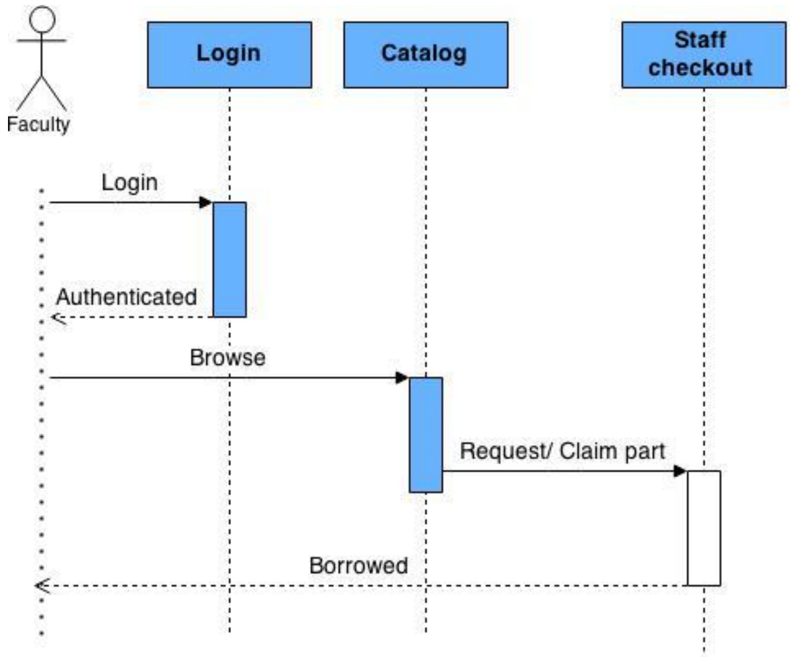


Figure 18 Faculty request sequence diagram

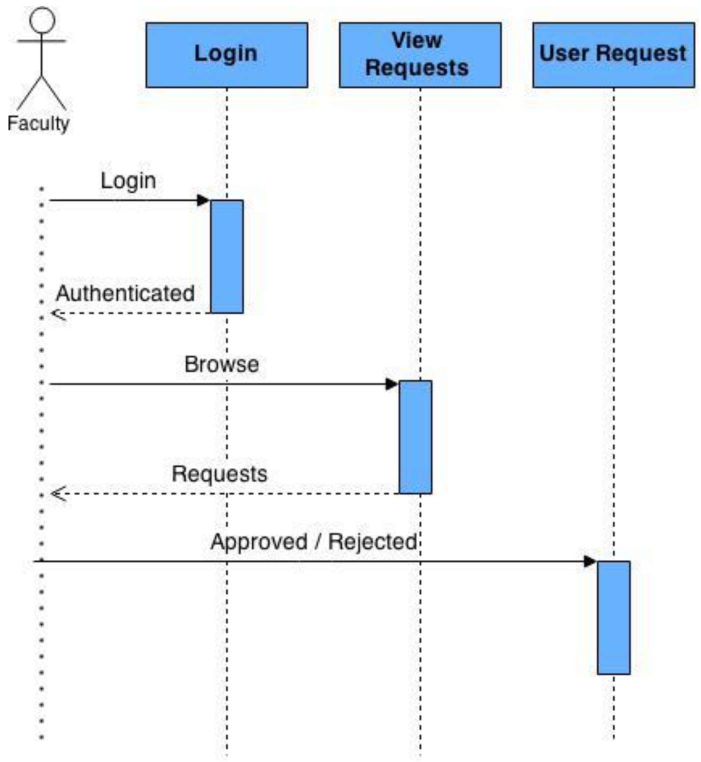


Figure 19 Faculty Approval sequence diagram

The **Staff** has four diagrams that show the sequence order of managing Categories, Parts, requesting parts and checkout requested parts. See Figure 20 to Figure 23 .

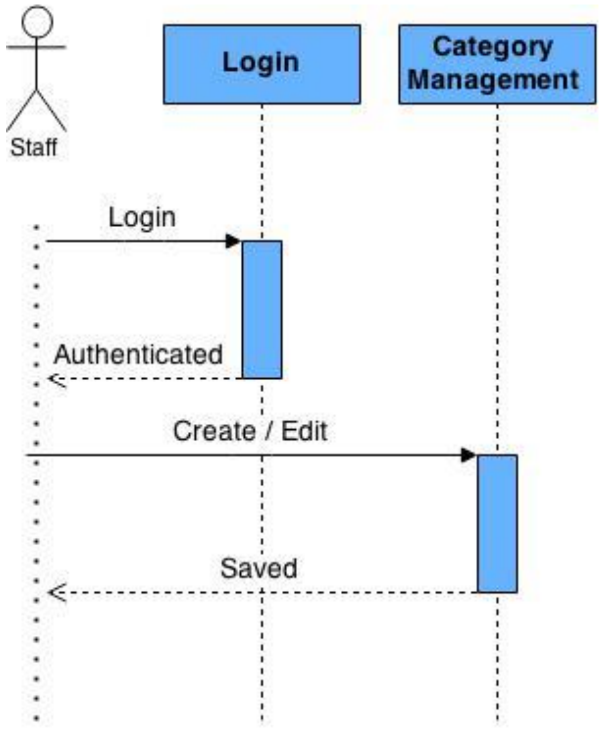


Figure 20 Staff category management sequence diagram

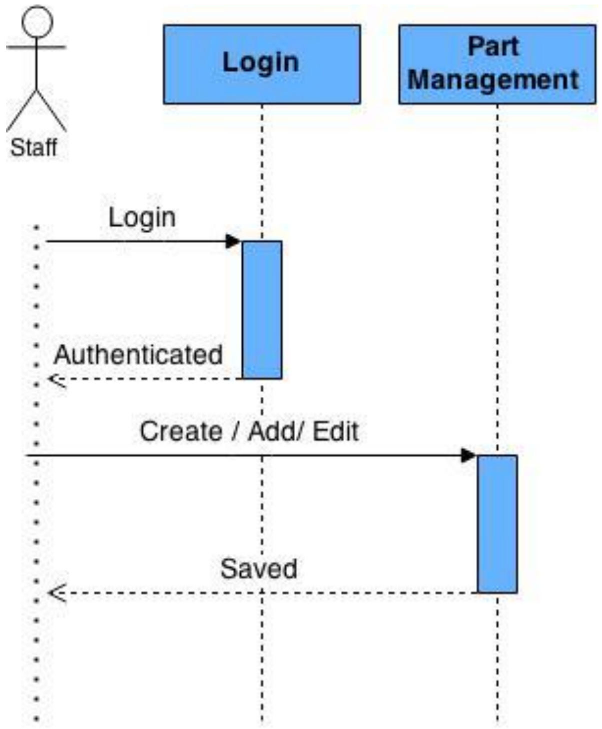


Figure 21 Staff Part Management sequence diagram

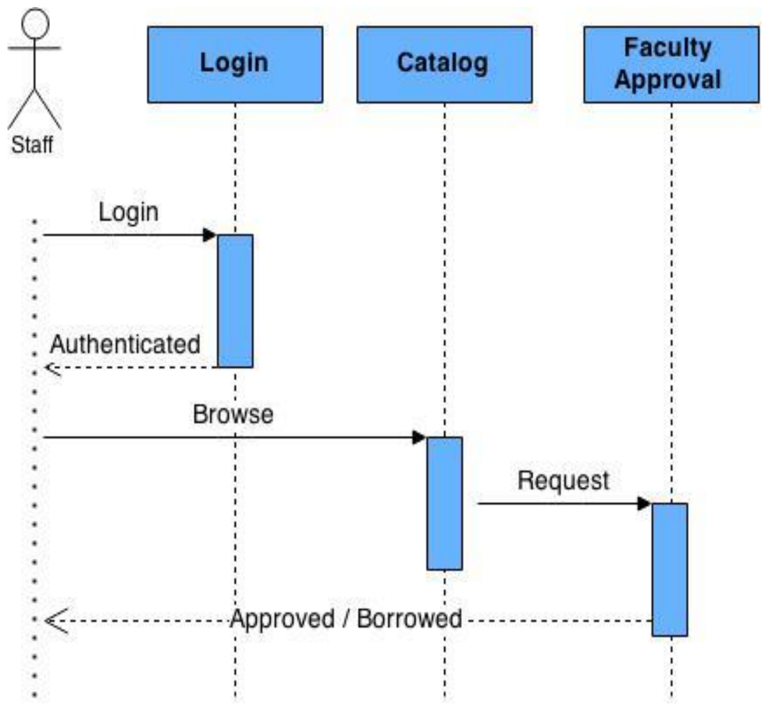


Figure 22 Staff request sequence diagram

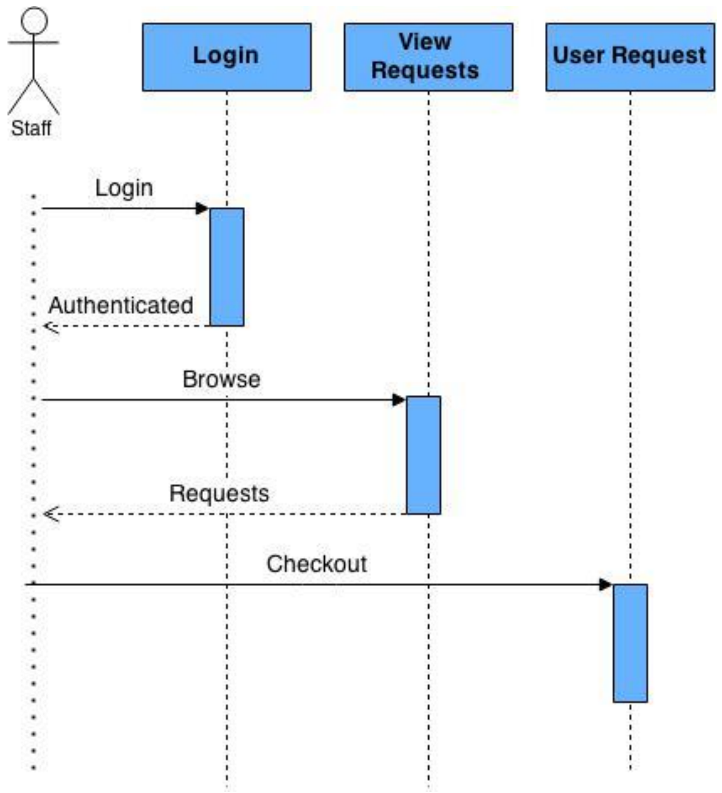


Figure 23 Staff checkout sequence diagram

The **Student** Sequence Diagram shows the sequence order of requesting parts.

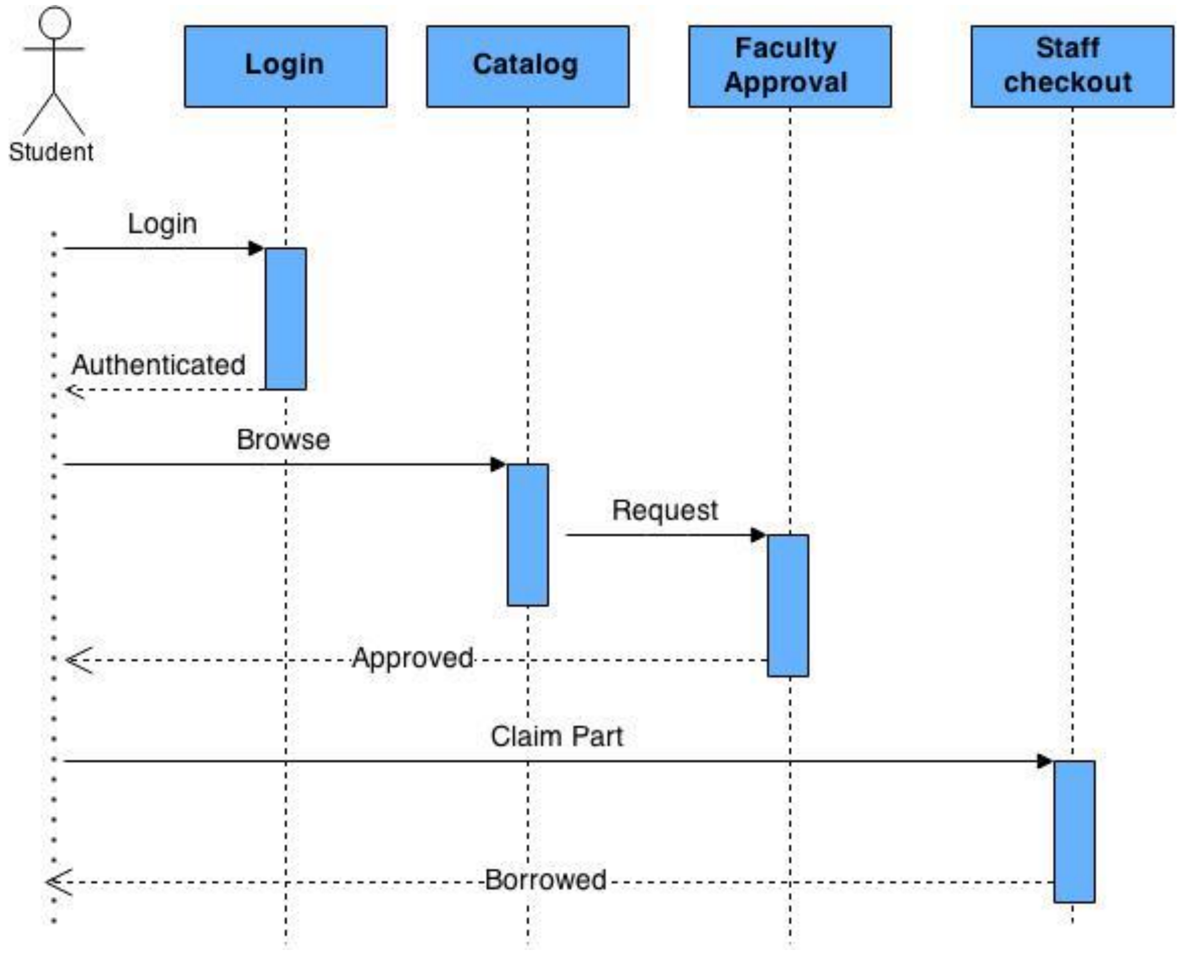


Figure 24 Student request sequence diagram

Activity Diagram

Activity Diagrams are used to show the system process as a flow of work through a series of actions. These actions can be described as start/end actions (rounded rectangle), processes (rectangle) and decisions (diamond).

The workflow of the **Admin** starts with User Management and ends with its activities.

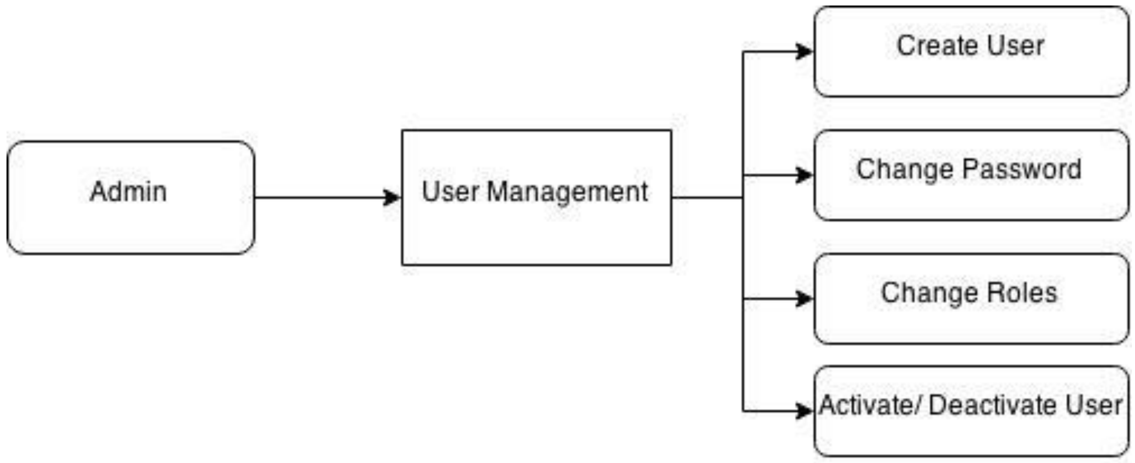


Figure 25 Admin activity diagram

The **Faculty** has two workflows. First, is requesting a part then waiting for Staff checkout. The other workflow is approve/decline the users' requests.

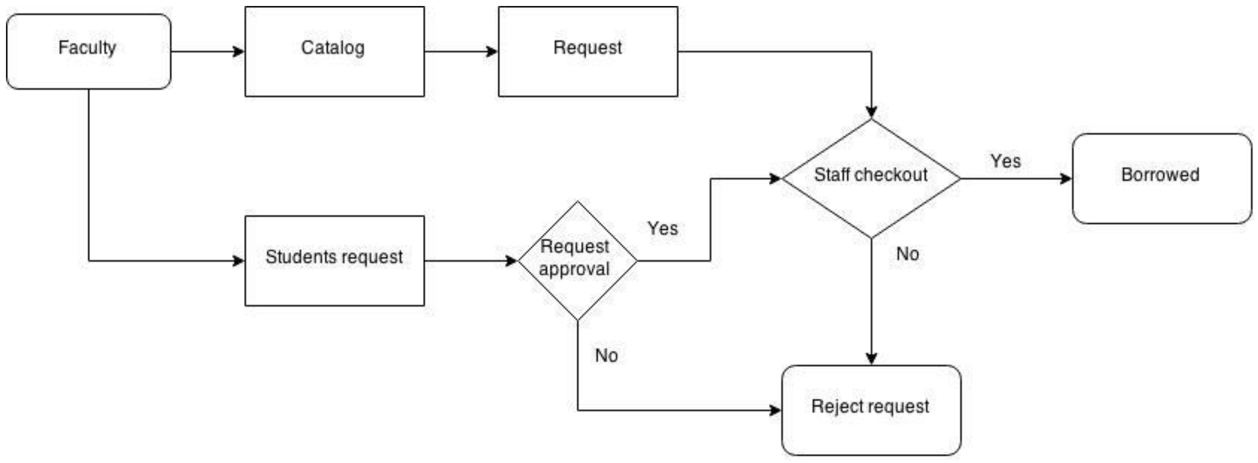


Figure 26 Faculty activity diagram

The **Staff** has three workflows which are, requesting parts from the system, checkout the approved requests and manage the Parts.

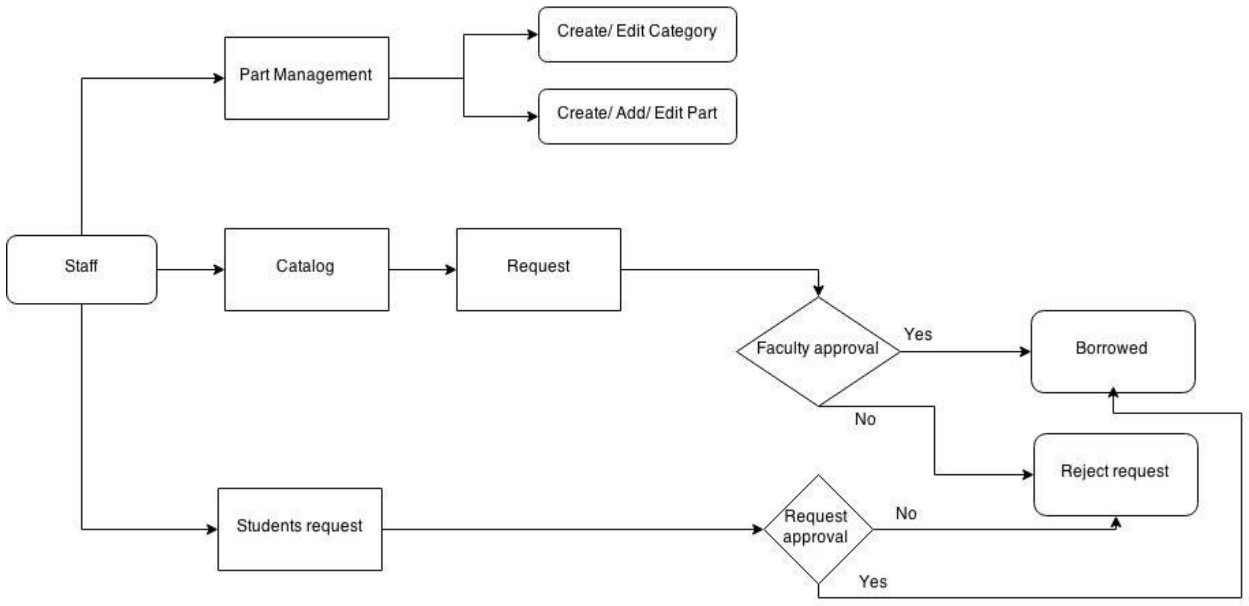


Figure 27 Staff activity diagram

The workflow of the **student** starts by making request, then faculty approval and staff checkout.

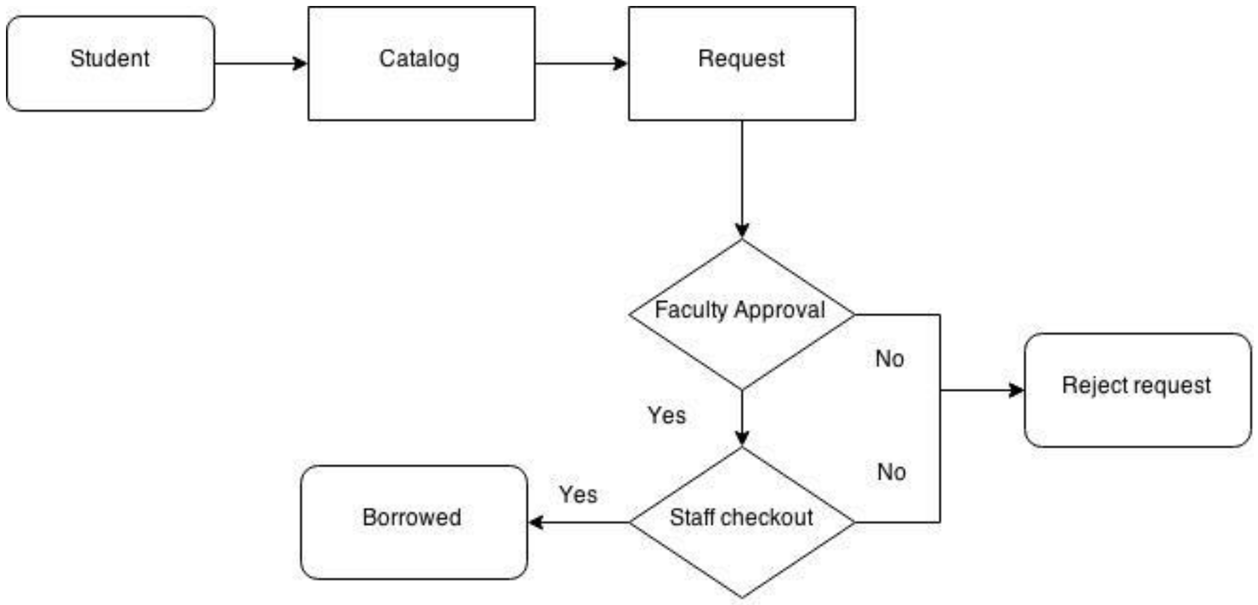


Figure 28 Student activity diagram

7. Deployment

There are multiple deployments options in regard of deploying Asp.NET web applications:

- XPS webserver (XML Paper Specification)
- IIS (Internet Information Service)
- Windows Azure (Cloud Service)

The team deployed the web application (Librarian) on IIS 8.5 on Windows Server 2012 R2, IIS 8.5 webserver is a feature of Windows Server 2012, and in order to deploy the system the team used the following IIS server configurations:

- Common HTTP Features
  - ✓ Static Content
  - ✓ Default Document
  - ✓ Directory Browsing
  - ✓ HTTP Errors

- Application Development
  - ✓ ASP.Net
  - ✓ .NET Extensibility
  - ✓ ISAPI Extensions
  - ✓ ISAPI Filters

- Health and Diagnostics
  - ✓ HTTP Logging
  - ✓ Request Monitor

- Security
  - ✓ Windows Authentication
  - ✓ Request Filtering

- Performance
  - ✓ Static Content Compression

- Management Tools
  - ✓ IIS Management Console

Master Data Services requires installation of the following features on the host server:

- .NET Framework 3.0 Features
  - ✓ WCF Activation
  - ✓ HTTP Activation
  - ✓ Non-HTTP Activation

- Windows PowerShell

- Windows Process Activation Service
  - ✓ Process Model
  - ✓ .NET Environment
  - ✓ Configuration APIs

The IIS default application pool identity setting needs to be configured as Network Services. Microsoft SQL Express database server was used to host the system database, and the database used in the system needs adds a user configured with username *'NT Authority\network Service'*.

System Initialization

When the system starts for the first time, some data needs to be present in the database like roles, groups and an Admin user. All of this is done automatically when the application starts through a special class called *AppStart*.cs. All what needs to be done for the system at its first start is a database connection added to the global *Web.config* file.

8. Testing, Analysis, and Evaluation

Testing methodology and results

The team tested the application using:

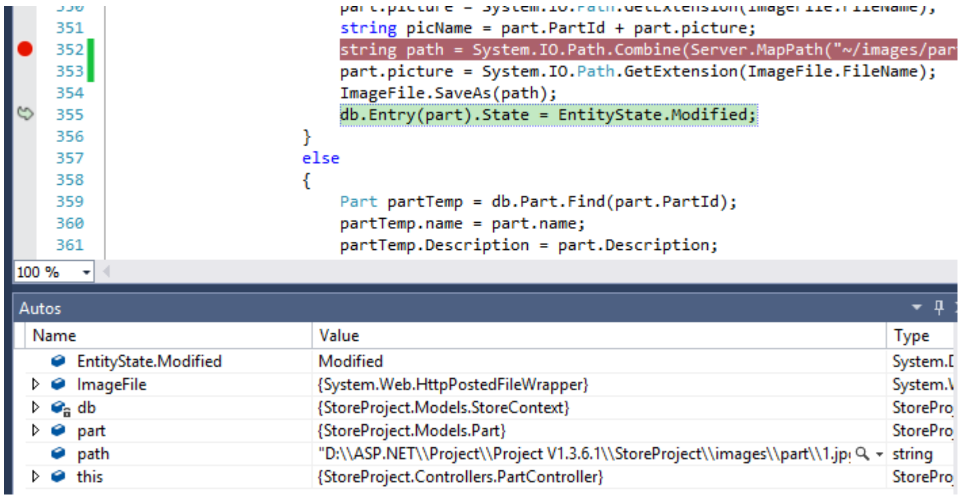
- **Unit testing**: testing each component alone.
- **Integration testing**: after integrating any components that integration get tested
- **System testing**: after deploying the complete system, it was put under tests to insure that system is working

The tests were performed by developing scenarios that are close as possible to the operational condition and apply them to the system and check whether the results comply with requirements.

Debugging

Finding the sources of errors was performed in multiple ways:

- **The IDE debugger**: Visual Studio provides a debugger to detect syntax errors.
- **Web Brewers**: while developing the system it was running on debug mode, so any sever errors will appear in the browser.
- **Break points**: this methodology is used to find the semantics errors where the code stops at the red point, and from there you can check the code instruction by instruction and the assigned values after each step. See Figure 29



```
            part.picture = System.IO.Path.GetExtension(ImageFile.FileName);
            string picName = part.PartId + part.picture;
            string path = System.IO.Path.Combine(Server.MapPath("~/images/par
            part.picture = System.IO.Path.GetExtension(ImageFile.FileName);
            ImageFile.SaveAs(path);
            db.Entry(part).State = EntityState.Modified;
        }
        else
        {
            Part partTemp = db.Part.Find(part.PartId);
            partTemp.name = part.name;
            partTemp.Description = part.Description;
```

Autos

Name	Value	Type
EntityState.Modified	Modified	System.
ImageFile	{System.Web.HttpPostedFileWrapper}	System.
db	{StoreProject.Models.StoreContext}	StorePro
part	{StoreProject.Models.Part}	StorePro
path	"D:\\ASP.NET\\Project\\Project V1.3.6.1\\StoreProject\\images\\part\\1.jp	string
this	{StoreProject.Controllers.PartController}	StorePro

Figure 29: Break Point debugging

System analysis and evaluation

As mentioned before using MVC design pattern will increase the efficiency, and performance. Security is also essential in the application.

Security

Since is essential to the system it's implemented in different parts of the system:

- **Web Browser – Server**: using the Secure Socket Shell (SSL) by generating self-certification to insure a secure connection between the browser and the web server.
- **Password in the database**: all the password stored in the database are hashed using libraries like *System.Security*;
- **Authentication and authorization**: the application is used by multiple users each with different privileges so the application has a login mechanism to insure authentication and privileges mechanisms implemented using "*Microsoft.AspNet.Identity;*" to insure authorization since each users group has different set of privilege, and the admin can configure which privileges to give to each user form the group's privileges that the user belongs to. See Figure 30

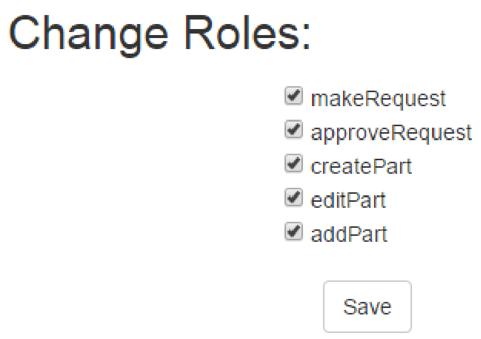


Figure 30: Roles configuring

9. Engineering Tools and Standards

Tools

After deciding the design pattern, now comes the choice of the frameworks. There are multiple frameworks for MVC, but the ones that stick out are:

- Play Framework (Java).
- Django Framework (Python).
- Asp.NET MVC.

Since the projects needed to be developed in a short time period, Django was excluded because the team doesn't have a background on Python programming, and using Python would have lengthened the learning curve. The team initial plan was to develop the application using Play Framework, but due to the poor documentation, and the Framework had a new release at the time of the development with major changes, the team decided to use Asp.NET MVC 5 with Razor templating engine, since it has a good documentation, it is easy to use and it serves the purpose. The IDE used in developing the application is Visual Studio 2013.

Standers

There are multiple communication protocols associated with web communication such as Http, FTP, TFTP. The application uses Https in the client-server communications, and if a user try to access the server using http, he will be redirected to https automatically.

10. Issues

Problems Faced

Developing the system the team encountered some issues and bugs, but the main challenging ones are:

- **Deployment of the system**:
    - **Description**: when the team tried to deploy the system, we encountered an internal server error 500.19 with code 0x80070021, that we didn't its description, and could not find a proper documentation that deals with the issue.
    - **Attempts**: the team tried deployed a new out of the box application but encountered the same error again.
    - **Solution**: after a lot of searching the team found that the issue is related to the server configurations, and solve the issue by configuring the server with configurations mentioned in the deployment section.

- **Session Bug**:
    - **Description**: when the admin changes a user privileges while this user is logged in, these changes will not take place until the user is logged of. Although, form the admin side these changes are applied, the user will still leverage these permission until he logs out.
    - **Attempts**: the team tried to force the user to logout if in case of privileges change, but the team didn't succeed implementing such a solution.
    - **Solution**: this bug was solved partially by adding timeout of 15 minutes for the session, if the application stay idle for 15 minutes the user will be logged out, by such a solution we insure that the user will eventually logout.

Limitations and constraints of the design

The design has a bottle neck because of the QR code scanner all request and return need to go through the staff in order to be scanned.

Limitations and constraints of the implementation

The application is implemented using Asp.NET MVC, and user management is implemented using the Identity Framework. This framework creates the user management database entities, these entities are shown previously in the schema figures (the ones with “ASP” word before the entity’s name), the limitation here is that these models (Entities) can only be changed, by extending them, and associating a foreign keys with them is really difficult, but there are two special classes (“*UserManger”* and “*RoleManger”)*, also provided by the Identity Framework that manages these models, and provides all the needed operations that needs to be done on these models.

In order to get more flexibility, the team created two database entities which are “*AppGroup*” and “*TempRoles*” (A replica of the “AspRoles”), this way manipulating the roles became easier and flexible. See Figure 31

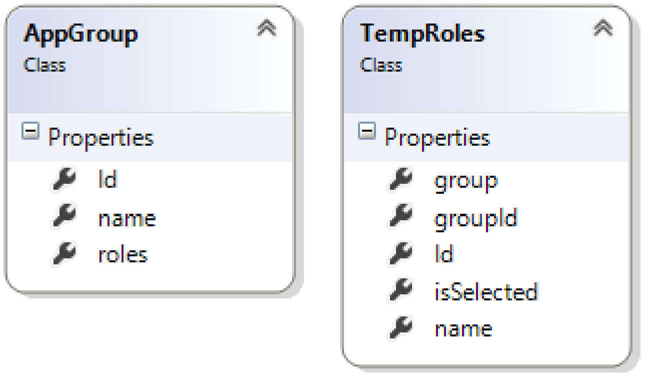


Figure 31 AppGroup and TempRoles Models

11. Conclusion

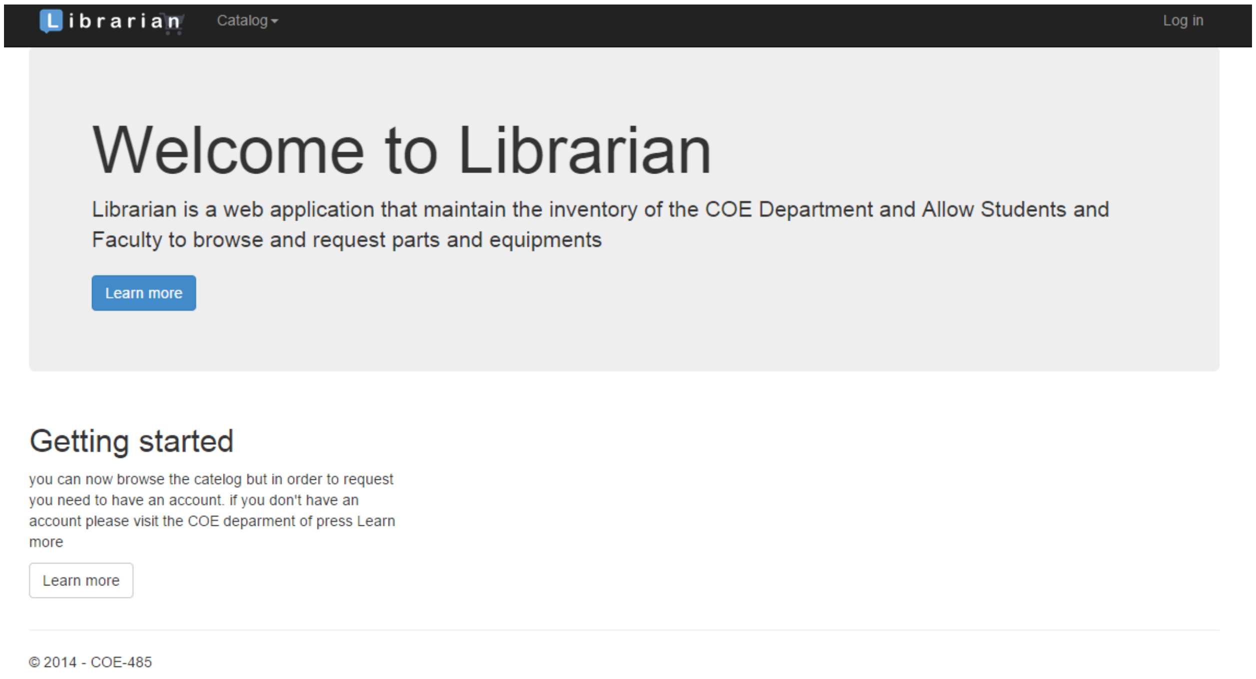
Developing this application gave the team members to gain a lot of skills associated with web development, standards and protocols. Also, experience a new field that involved many engineering concepts and standers. If there was more time, the team would have implemented the project using different framework, such as Django or Play. The team member learning outcomes are listed below:

- Teamwork
- Web development
- MVC design pattern
- Object Relation Mapper (ORM)
- C# programing language
- HTML markup language
- JQuery
- CSS styling language
- Java scripts
- Implementing security techniques
- Interfacing Hardware components with web application
- Web application deployment
- Web Server configuration

12. References

- Connections for Corded Barcode Scanners. (n.d.). Retrieved December 20, 2014, from http://www.carolinabarcode.com/barcode-scanner-connections-a-67.html
- Diepenmaat, J. (n.d.). The Model View Controller pattern in web applications. Retrieved December 20, 2014, from http://zeekat.nl/articles/mvc-for-the-web.html
- SQL Server Answers. (n.d.). Retrieved December 20, 2014, from http://go4answers.webhost4life.com/Example/mds-setup-http-error-50019-internal-16458.aspx
- ASP.NET MVC Overview. (n.d.). Retrieved December 20, 2014, from http://msdn.microsoft.com/en-us/library/dd381412(v=vs.108).aspx

13. Appendix A: Some Screen Shots of the Application



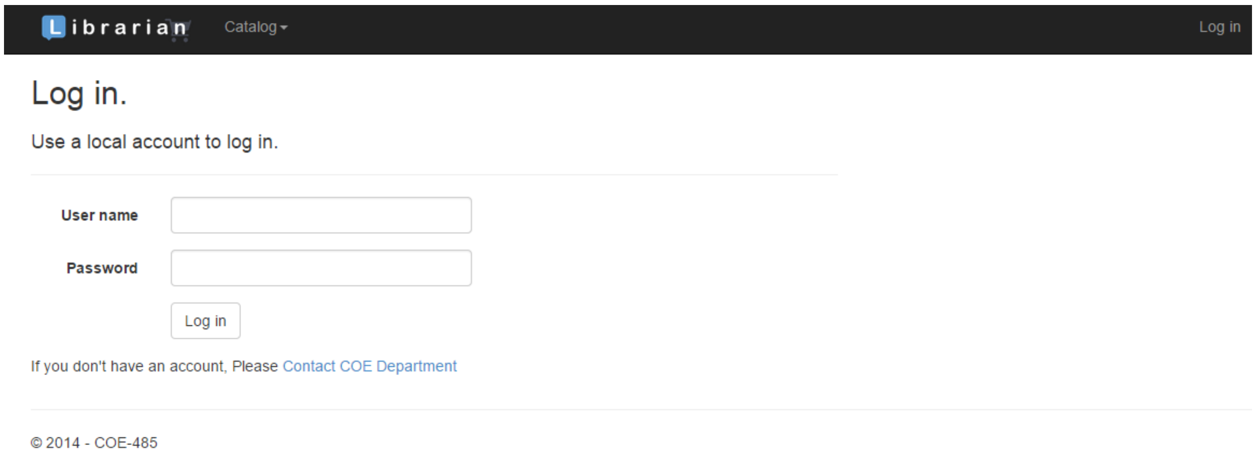
Welcome to Librarian

Librarian is a web application that maintain the inventory of the COE Department and Allow Students and Faculty to browse and request parts and equipments

Getting started

you can now browse the catelog but in order to request you need to have an account. if you don't have an account please visit the COE deparment of press Learn more

© 2014 - COE-485



Log in.

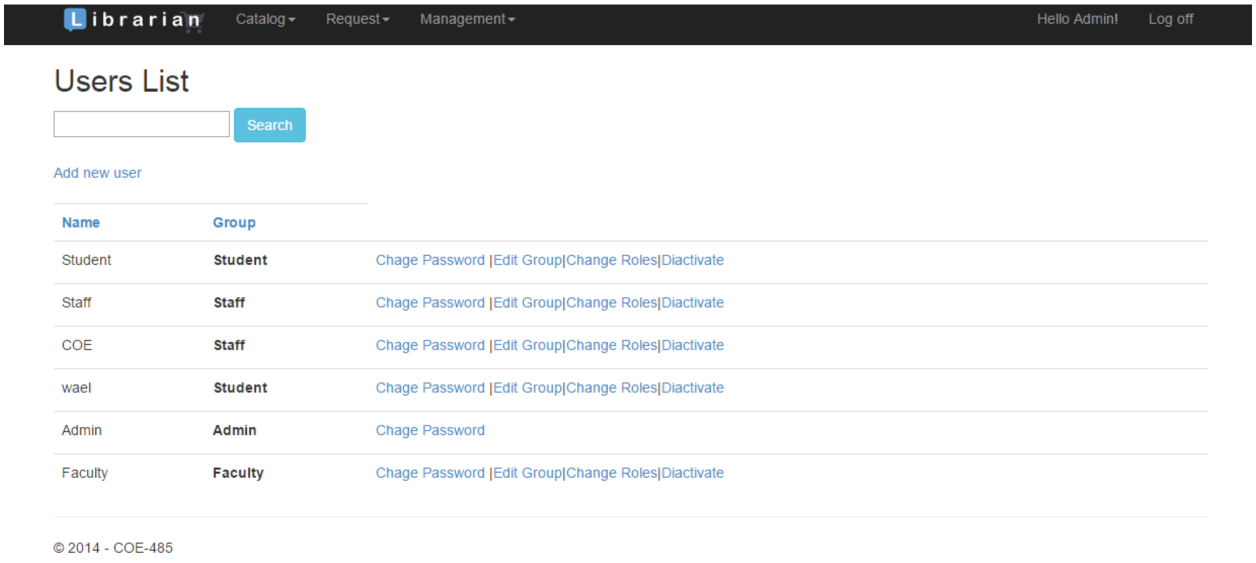
Use a local account to log in.

User name

Password

If you don't have an account, Please Contact COE Department

© 2014 - COE-485



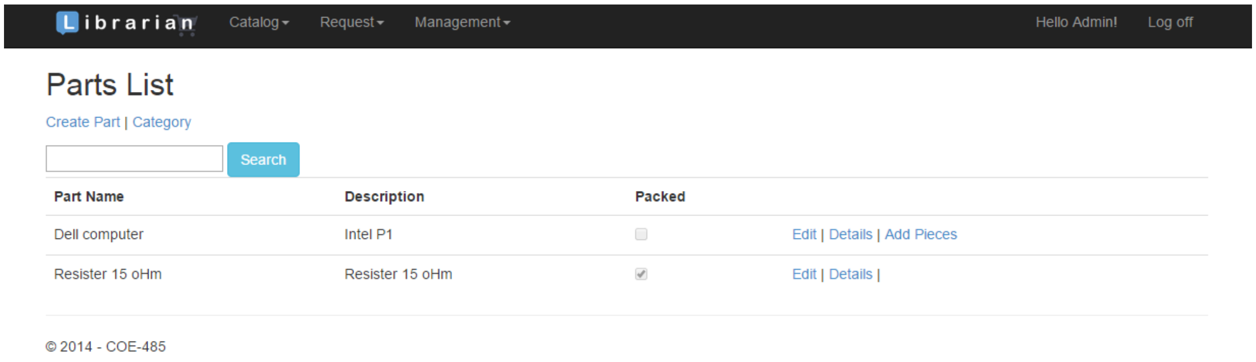
Users List

Search

Add new user

Name	Group	
Student	Student	Chage Password |Edit Group|Change Roles|Diactivate
Staff	Staff	Chage Password |Edit Group|Change Roles|Diactivate
COE	Staff	Chage Password |Edit Group|Change Roles|Diactivate
wael	Student	Chage Password |Edit Group|Change Roles|Diactivate
Admin	Admin	Chage Password
Faculty	Faculty	Chage Password |Edit Group|Change Roles|Diactivate

© 2014 - COE-485



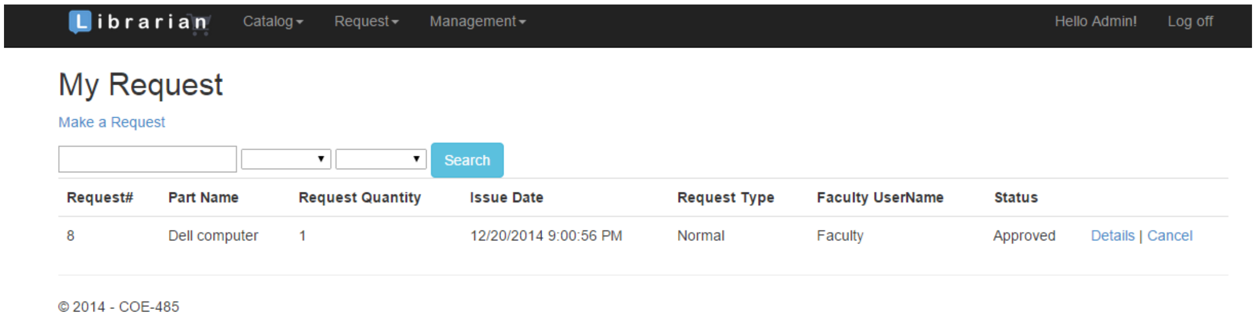
Parts List

Create Part | Category

Search

Part Name	Description	Packed	
Dell computer	Intel P1	☐	Edit | Details | Add Pieces
Resister 15 oHm	Resister 15 oHm	✓	Edit | Details |

© 2014 - COE-485



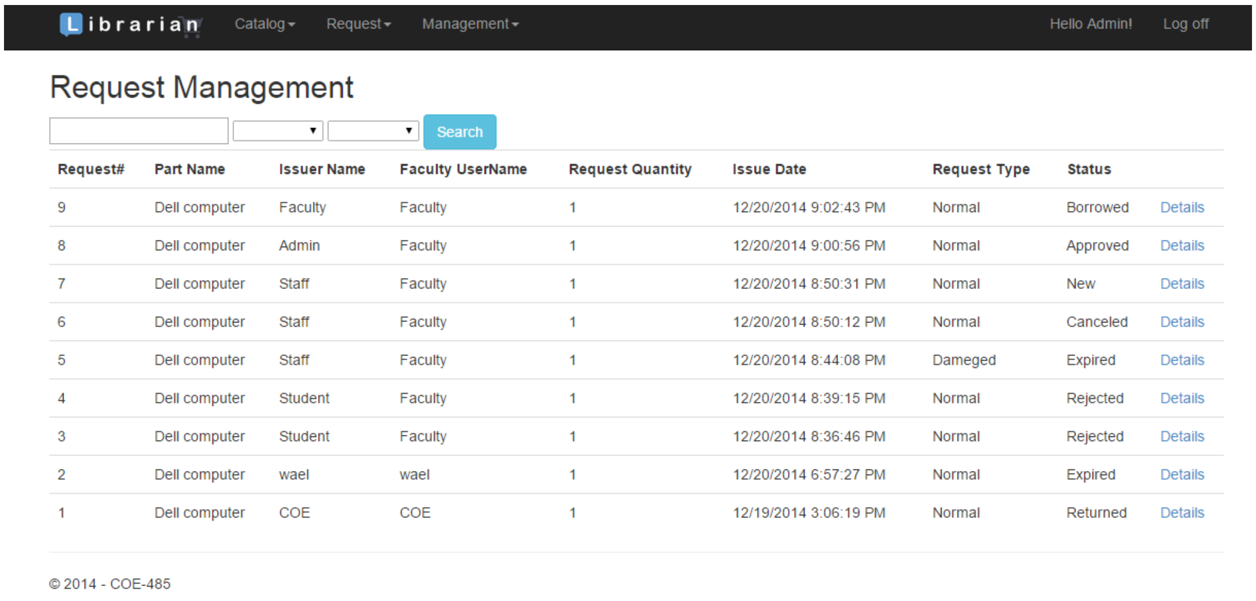
My Request

Make a Request

Search

Request#	Part Name	Request Quantity	Issue Date	Request Type	Faculty UserName	Status	
8	Dell computer	1	12/20/2014 9:00:56 PM	Normal	Faculty	Approved	Details | Cancel

© 2014 - COE-485

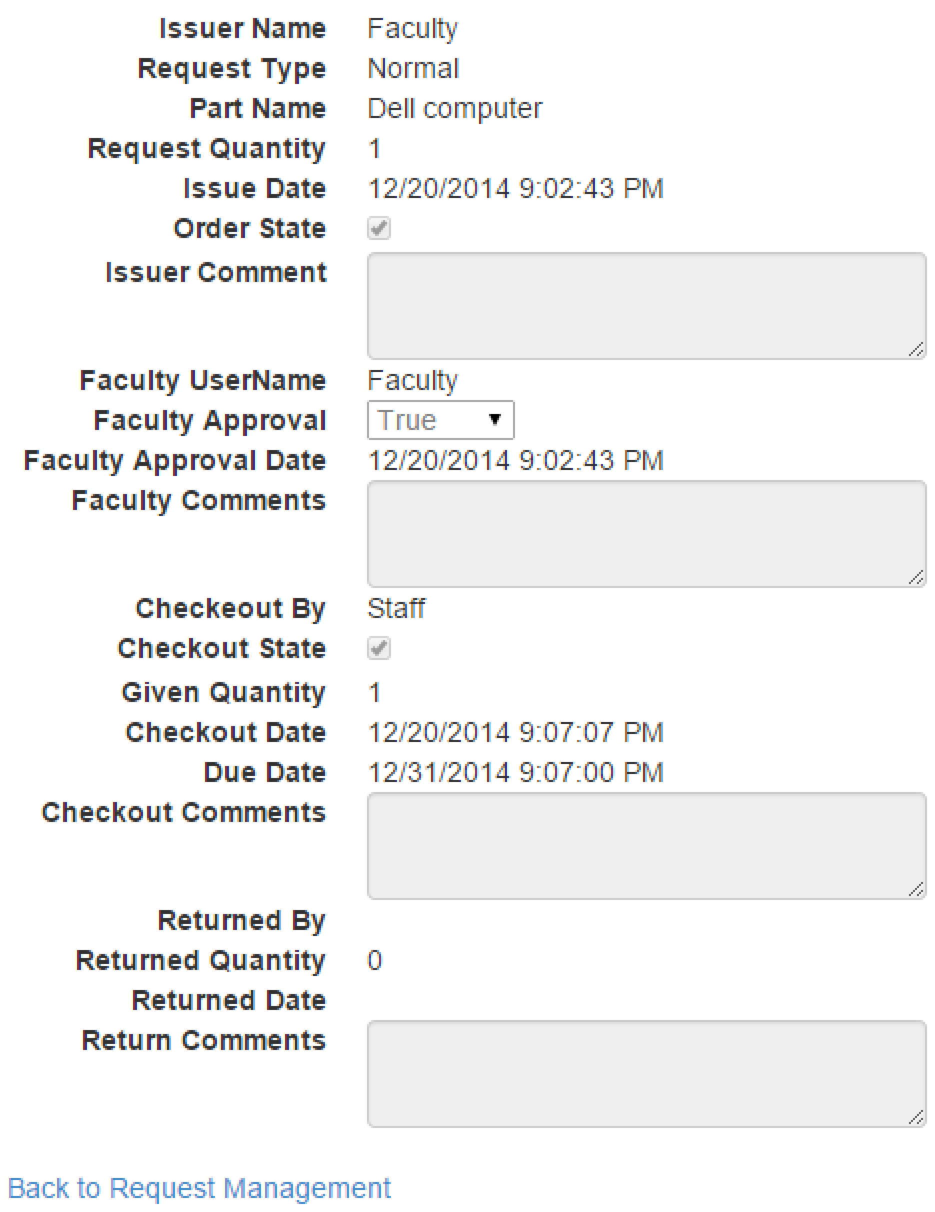

Librarian Catalog Request Management Hello Admin! Log off

Request Management

Search

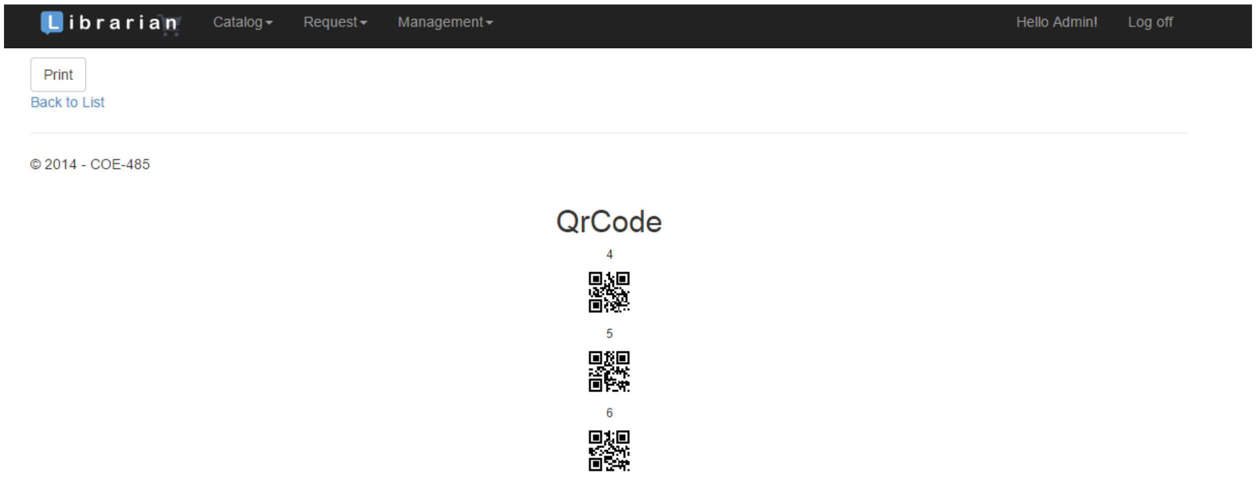
Request#	Part Name	Issuer Name	Faculty UserName	Request Quantity	Issue Date	Request Type	Status	
9	Dell computer	Faculty	Faculty	1	12/20/2014 9:02:43 PM	Normal	Borrowed	Details
8	Dell computer	Admin	Faculty	1	12/20/2014 9:00:56 PM	Normal	Approved	Details
7	Dell computer	Staff	Faculty	1	12/20/2014 8:50:31 PM	Normal	New	Details
6	Dell computer	Staff	Faculty	1	12/20/2014 8:50:12 PM	Normal	Canceled	Details
5	Dell computer	Staff	Faculty	1	12/20/2014 8:44:08 PM	Dameged	Expired	Details
4	Dell computer	Student	Faculty	1	12/20/2014 8:39:15 PM	Normal	Rejected	Details
3	Dell computer	Student	Faculty	1	12/20/2014 8:36:46 PM	Normal	Rejected	Details
2	Dell computer	wael	wael	1	12/20/2014 6:57:27 PM	Normal	Expired	Details
1	Dell computer	COE	COE	1	12/19/2014 3:06:19 PM	Normal	Returned	Details

© 2014 - COE-485



Issuer Name	Faculty
Request Type	Normal
Part Name	Dell computer
Request Quantity	1
Issue Date	12/20/2014 9:02:43 PM
Order State	✓
Issuer Comment	
Faculty UserName	Faculty
Faculty Approval	True
Faculty Approval Date	12/20/2014 9:02:43 PM
Faculty Comments	
Checkeout By	Staff
Checkout State	✓
Given Quantity	1
Checkout Date	12/20/2014 9:07:07 PM
Due Date	12/31/2014 9:07:00 PM
Checkout Comments	
Returned By	
Returned Quantity	0
Returned Date	
Return Comments	

Back to Request Management



Librarian Catalog Request Management Hello Admin! Log off

Print

Back to List

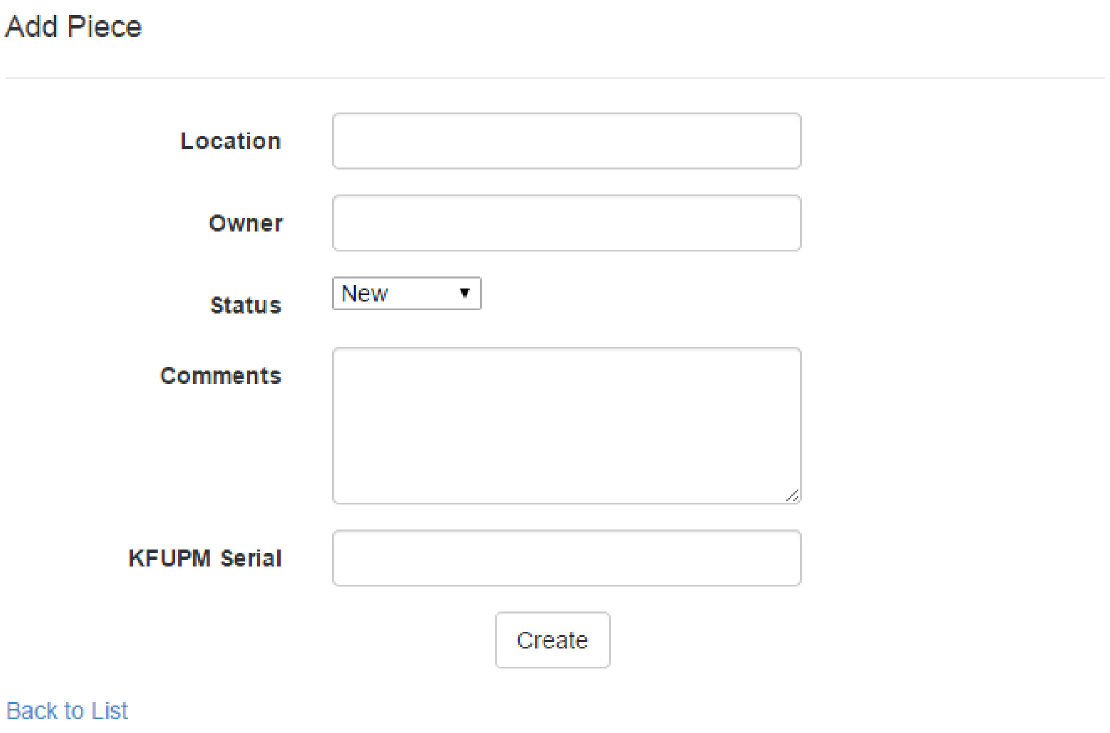
© 2014 - COE-485

QrCode

4

5

6



Add Piece

Location

Owner

Status New

Comments

KFUPM Serial

Create

Back to List