March 2022
Ver 2.5

HENRY BOOKS INVENTORY SYSTEM

ITEC 6420 DISTRIBUTED ENTERPRISE SOFTWARE DESIGN AND DEVELOPMENT

Architecture and Design document

FINAL PROJECT: GANESH SARGAM

REVISION HISTORY

Date	Author(s)	Version	Change Reference
02/25/2023	Ganesh Sargam	1.0	Initial Design
02/27/2023	Ganesh Sargam	1.1	Added drft usecases
02/28/2023	Ganesh Sargam	1.2	Updated usecases, sequence, ui desgn and other scenaiores
03/01/2023	Ganesh Sargam	1.4	Added initial technical design diagrams
03/02/2023	Ganesh Sargam	1.7	Review comments updated, Added class diagrams , CRC diagrams
03/03/2023	Ganesh Sargam	1.9	Added applicaton architecture and other technical design diagrams
03/04/2023	Ganesh Sargam	2.0	Added/updated technical details
03/05/2023	Ganesh Sargam	2.5	Final review done

Contents

I. INTRODUCTION

This document provides a comprehensive architectural overview of the system, using a number of different architectural views to depict different aspects of the system. It is intended to capture and convey the significant architectural decisions which have been made on the system.

1.1. PURPOSE

This document outlines the architectural overview of the system **Mobile Online Henry's Books Inventory System**.

1.2. SCOPE

This Software Architecture Document provides an architectural overview of the Mobile Online Henry's Books Inventory. The Mobile Online Henry's Books Inventory is being developed by Ganesh Sargam to support Henry's Book Inventory Management

The system is mainly for the staff and students at the university. The intended audience of this document are:

- **Staff members** – They will access the university registration portal and review the courses and students' registrations.
- **Students** – They will access the university registration portal and review the courses and respective schedules.

1.3. OVERVIEW

The system will let Henry's chain to manage book's inventory using mobile android app and his chain including branches can add book and branch information to maintain inventory. This app also useful to view final report to view inventory of the books by branch.

II. REFERENCES

Applicable references are:

- SoftwareRequirementsSpecification.pdf
- ITEC 6420 Group 3: Final Project (3/5) Middle Georgia State University Final Assignment
- ITEC 6420 Distributed Enterprise Software Design and Development – Online PDF document (mga.view.usg.edu)

III. ARCHITECTURAL REPRESENTATION

This document presents the architecture as a series of views; use case view, logical view, process view and deployment view. There is no separate implementation view described in this document. These are views on an underlying Unified Modeling Language (UML) model developed using draw.io.

IV. ARCHITECTURAL GOALS AND CONSTRAINTS

Henry is the owner of a bookstore chain named Henry Books. Henry wants to manage book inventory of each chain using a mobile app. The system is mainly for the Henry and his chain (including all the branches) to maintain and view Book Inventory.

- Enable the branches to add branch information
- Allow to add book information and availability information by branch
- Should have option to view inventory report

- Database server (MySQL is recommended) should be hosted on a separate Google VM instance. So, use two Google VM instances; one for REST API and another for database server.
- REST API should run on a Google or other Cloud VM instance
- Need to create a firewall rule to allow communications between two servers (database and REST API)
- REST API will be implemented including at least the following three entities:
    - Book: This entity records information about the books that are being sold in the bookstore chain

      Properties:

      Id (int),

      Author (string),

      Title (string),

      Description (string),

      ThumbnailUrl (string),

      Price (double)

- branch: This entity records information about branches of the bookstore chain.

  Properties:

  Id (int),
  BranchName (string),
  Address (string),
  City (string),
  State (string),
  Zip (string),
  Phone (string)

- inventory: This entity records book inventory of each branch. Note that book's Id and branch's Id are foreign keys.

  Properties:

  Id (int),

  BookId (int),

  BranchId (int),

  quantity (int)

- REST API should run on a Google or other Cloud VM instance.
- REST API should be able to do the CRUD operations (i.e., select, insert, update, and delete) against the three entities (i.e., database tables)
- Database server (MySQL is recommended) should be hosted on a separate Google VM instance. So, use two Google VM instances; one for REST API and another for database server. You will need to create a firewall rule to allow communications between two servers.
- REST API should be implemented using Java Spring Boot and MVC framework.
- Android app and REST API should communicate using JSON messages to do the CRUD operations.

V. USE-CASE DESCRIPTION – ADD BOOK

- **Brief Description**
  A description of the use-case view of the software architecture. Adding book to the system

  - **Flow of Events**

    Add new book to the Henry's Book Store Android app use cases are:

    - Open Android app called Henry's Book Store

    - System will open landing page with bunch of buttons

    - Click (touch) to open "Add Book"

    - System will navigate to Screen to input book author, Title, Description, ThumbnailUrl and Price of the book

    - Once input is done click on "Add New Book" button

    - System will show "Successfully added!" message

- **Preconditions**

  - Before this use case begins the user should have Android phone and installed this app on his phone, user installation manual will be provided later.

VI. USE-CASE DESCRIPTION – ADD BRANCH

- **Brief Description**
  A description of the use-case view of the software architecture. Adding BRANCH to the system

  - **Flow of Events**

    Add new branch to the Henry's Book Store Android app use cases are:

    - Open Android app called Henry's Book Store

    - System will open landing page with bunch of buttons

    - Click (touch) to open "Add Branch"

    - System will navigate to Screen to input Branch Name, Address, City, State, Zip Code and Phone number of the branch

    - Once input is done click on "Add Branch" button

    - System will show "Successfully added!" message

- **Preconditions**

  - Before this use case begins the user should have Android phone and installed this app on his phone, user installation manual will be provided later.

VII. USE-CASE DESCRIPTION – ADD INVENTORY

- **Brief Description**
  A description of the use-case view of the software architecture. Adding INVENTORY to the system

  - **Flow of Events**

    Add Inventory to the Henry's Book Store Android app use cases are:

    - Open Android app called Henry's Book Store

    - System will open landing page with bunch of buttons

    - Click (touch) to open "Add Inventory" button

    - System will navigate to Screen to input Book ID(existing), Branch ID(existing) and available quantity at the branch

    - Once input is done click on "Add" button

    - System will show "Successfully added!" message

- **Preconditions**

  - Before this use case begins the user should have Android phone and installed this app on his phone, user installation manual will be provided later.

VIII. USE-CASE DESCRIPTION – VIEW INVENTORY REPORT

- **Brief Description**
  A description of the use-case view of the software architecture. View INVENTORY Report

  - **Flow of Events**

    To View Inventory Report from the Henry's Book Store Android app use cases are:

    - Open Android app called Henry's Book Store

    - System will open landing page with bunch of buttons

    - Click (touch) to open "Inventory Report" button

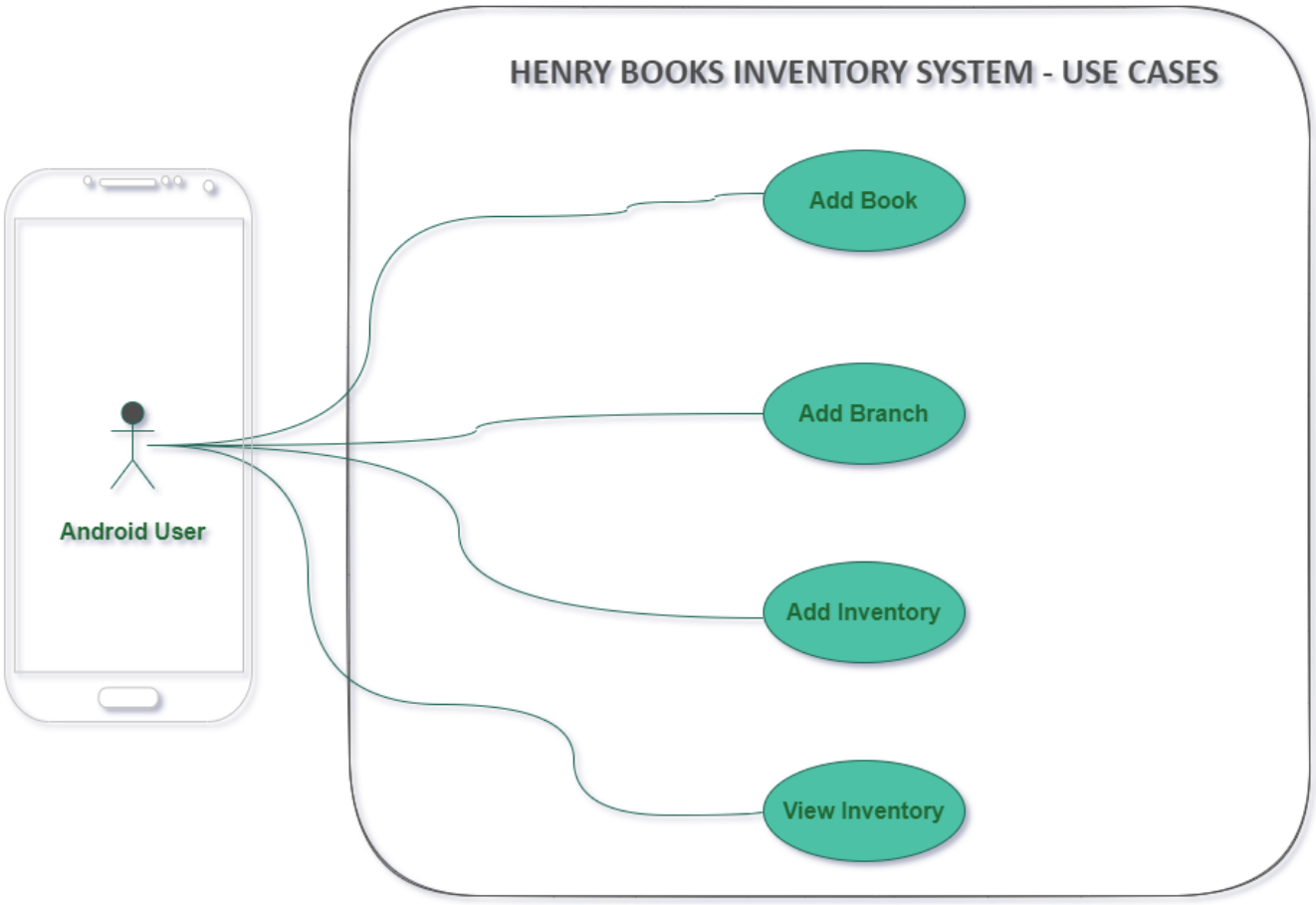
    - System will navigate to Screen and load Inventory report in the tabular format and should have following columns:

      Book Id, and associated Branch Id and the available quantity by branch

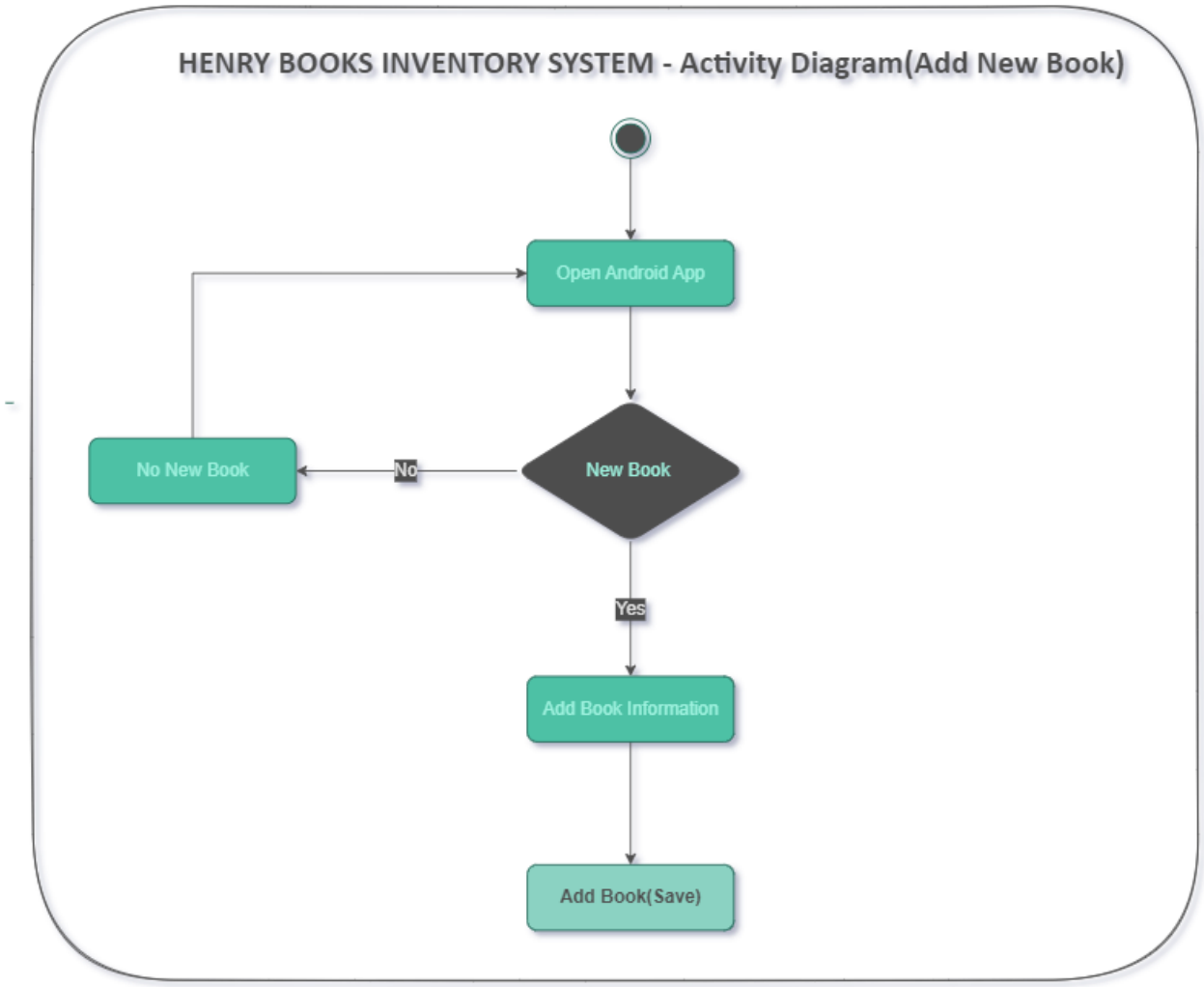
- **Preconditions**

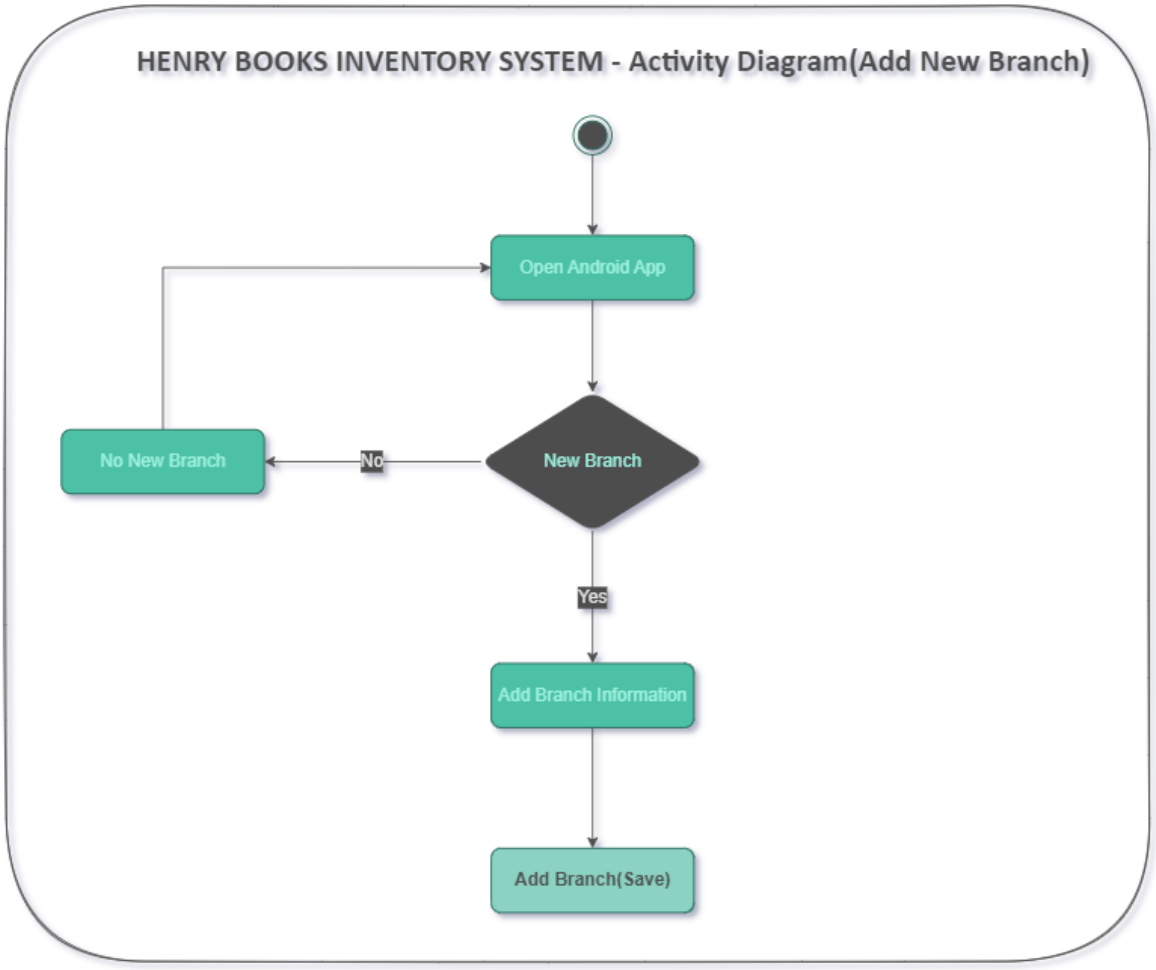
  - Before this use case begins the user should have Android phone and installed this app on his phone, user installation manual will be provided later.

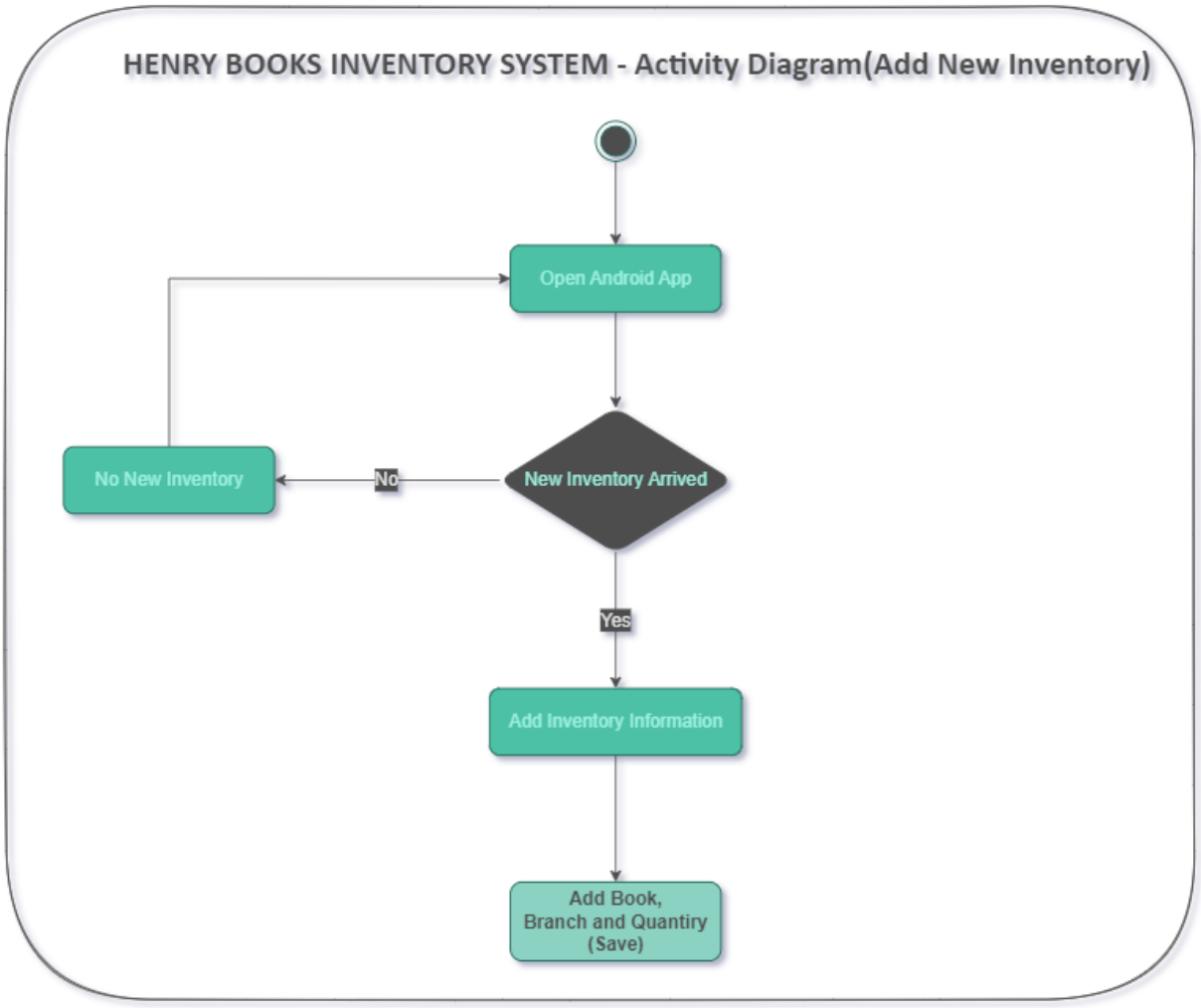
IX. ARCHITECTURALLY-SIGNIFICANT USE CASES

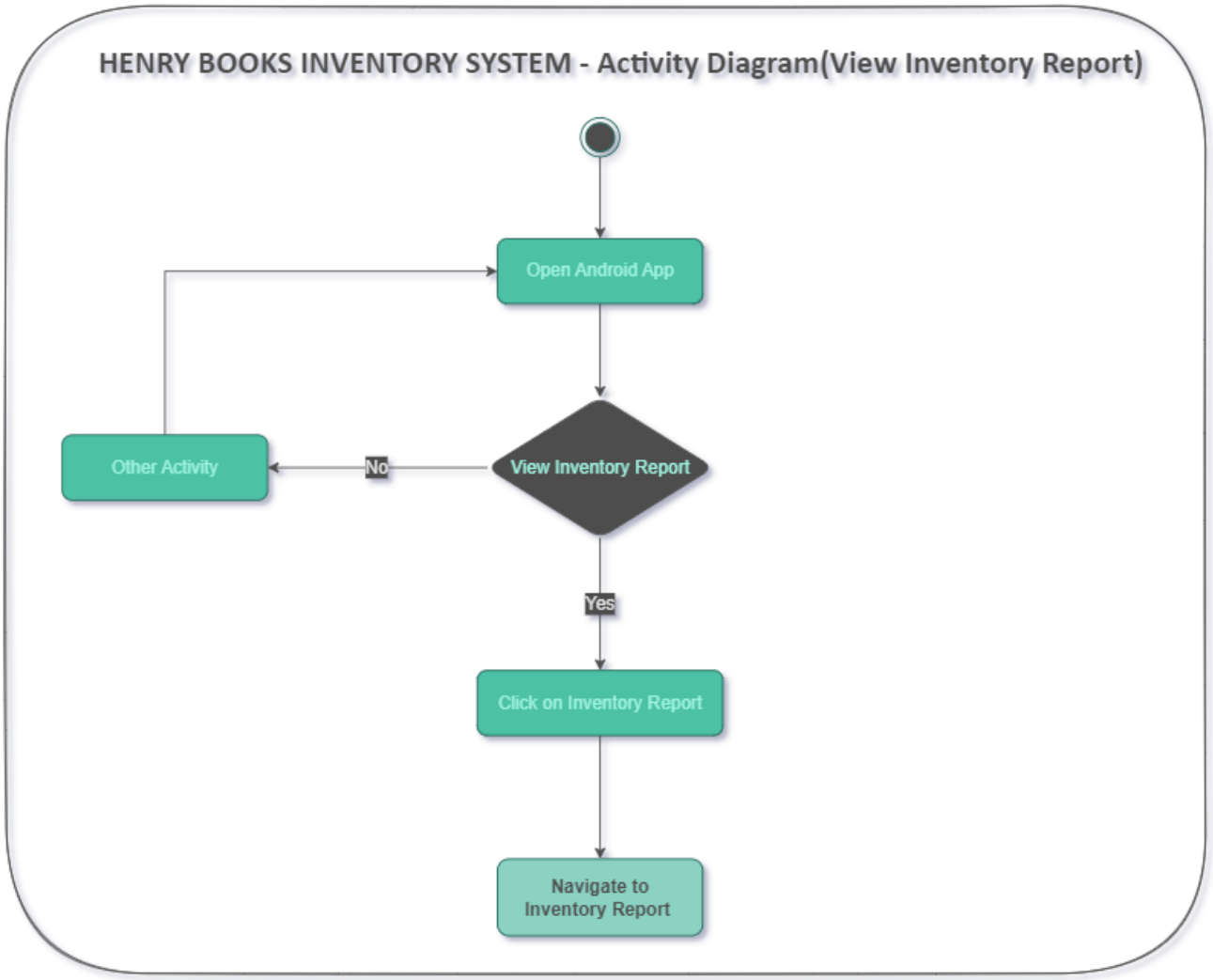


- ACTIVITY DIAGRAM




HENRY BOOKS INVENTORY SYSTEM - Activity Diagram(Add New Branch)
Open Android App
New Branch
No
No New Branch
Yes
Add Branch Information
Add Branch(Save)


HENRY BOOKS INVENTORY SYSTEM - Activity Diagram(Add New Inventory)
Open Android App
New Inventory Arrived
No
No New Inventory
Yes
Add Inventory Information
Add Book,
Branch and Quantity
(Save)

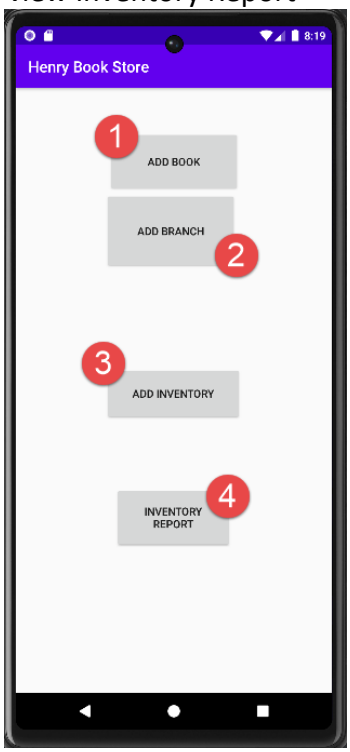

HENRY BOOKS INVENTORY SYSTEM - Activity Diagram(View Inventory Report)
Open Android App
View Inventory Report
No
Other Activity
Yes
Click on Inventory Report
Navigate to Inventory Report

- UI DESIGNS

❖ Android App Home

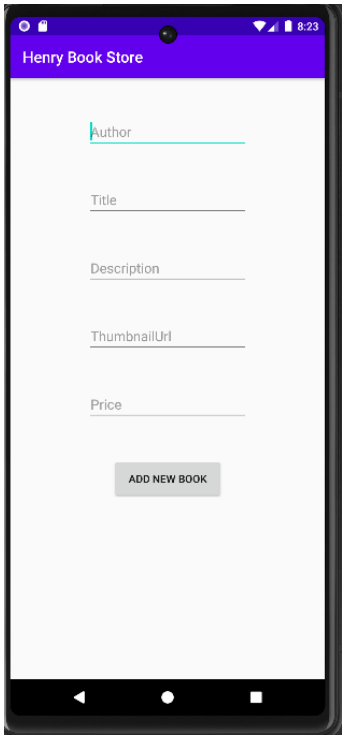
❖ Landing Screen:

- ♦ Add Book Button
- ♦ Add Branch Button
- ♦ Add Inventory
- ♦ Button to view Inventory Report



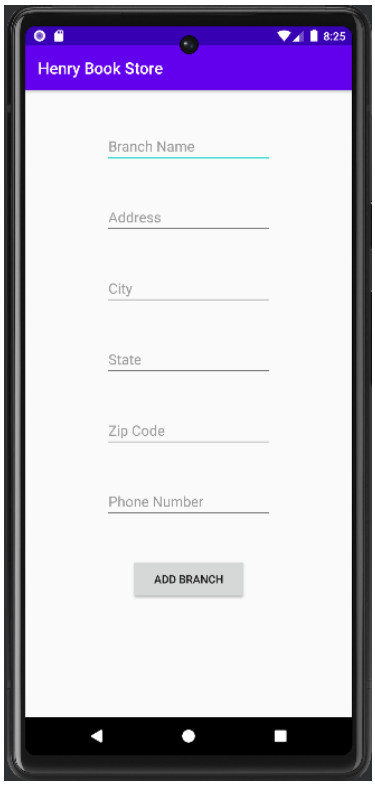
❖ Add Book into the system:

:

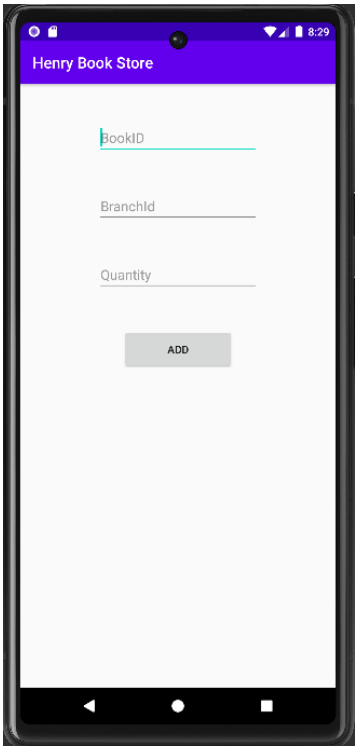

8:23
Henry Book Store
Author
Title
Description
ThumbnailUrl
Price
ADD NEW BOOK

❖ Add Branch into the system:

:

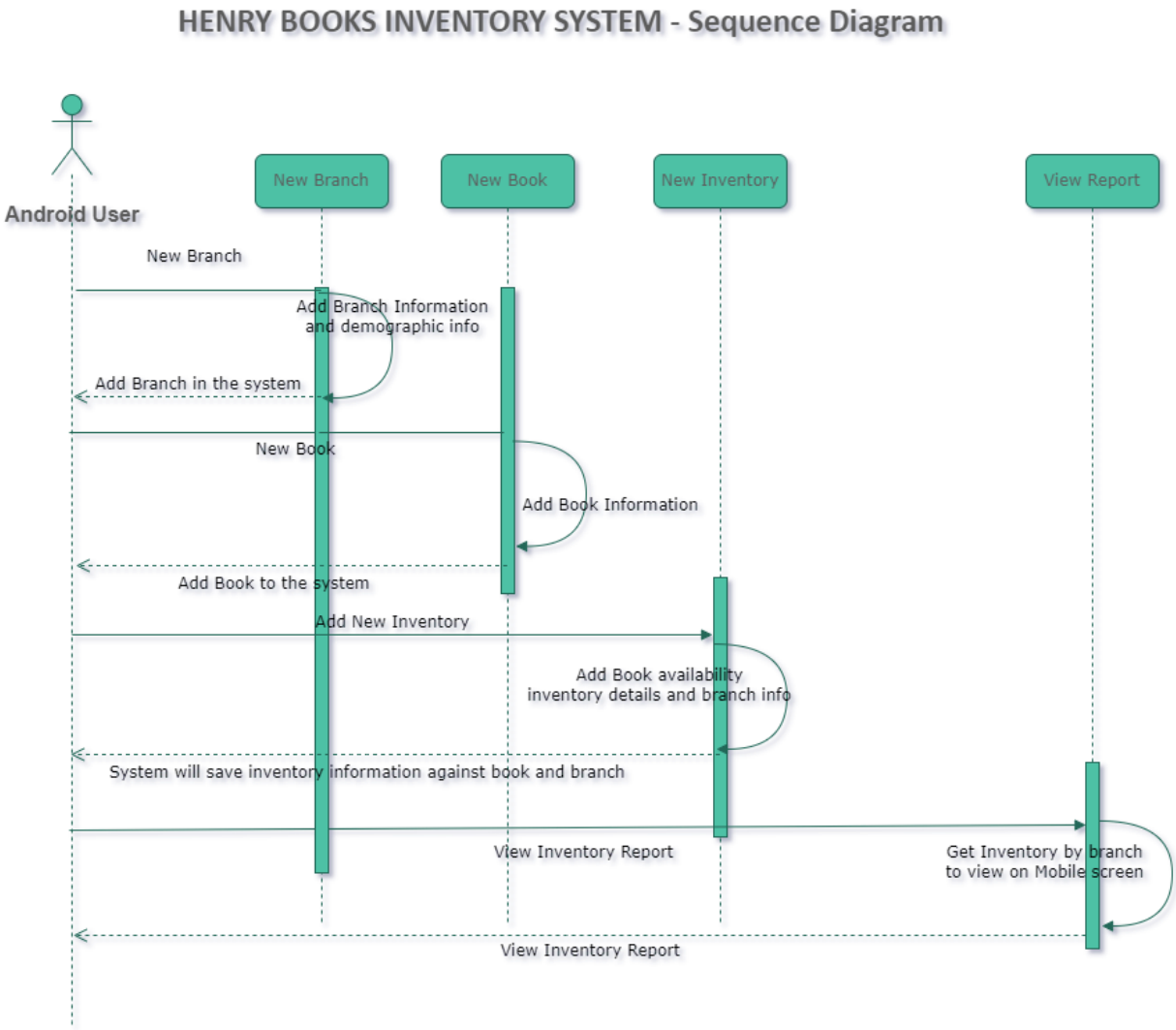


- Add Inventry into the system:

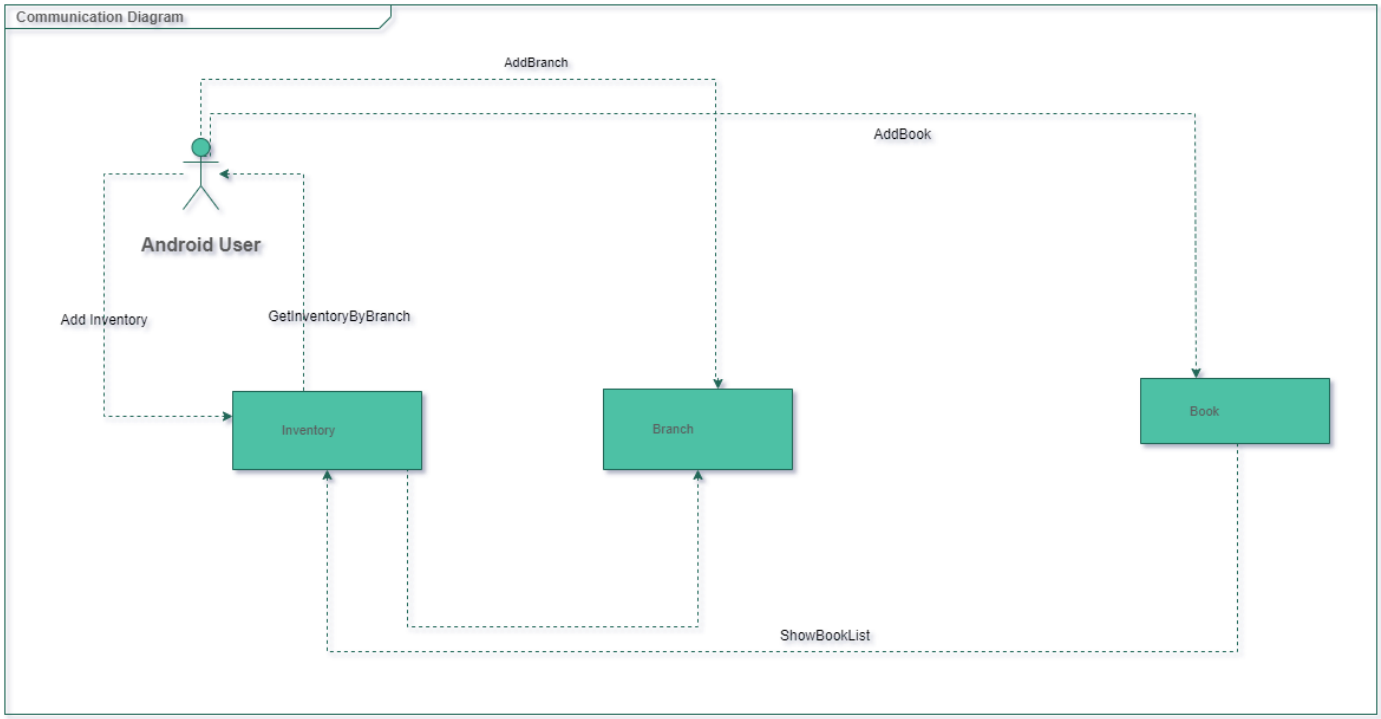


- Inventory Report View:

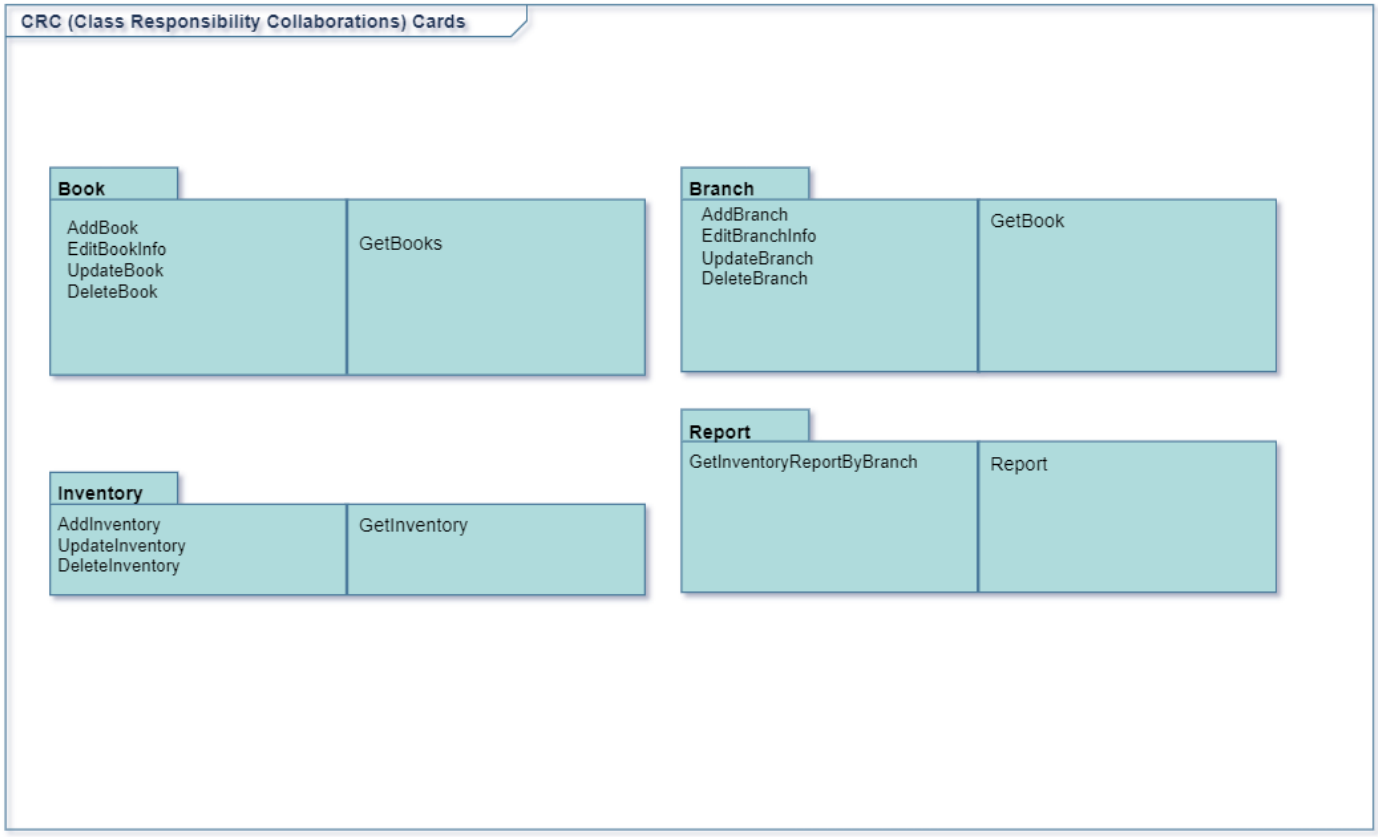
- SEQUENCE DIAGRAM



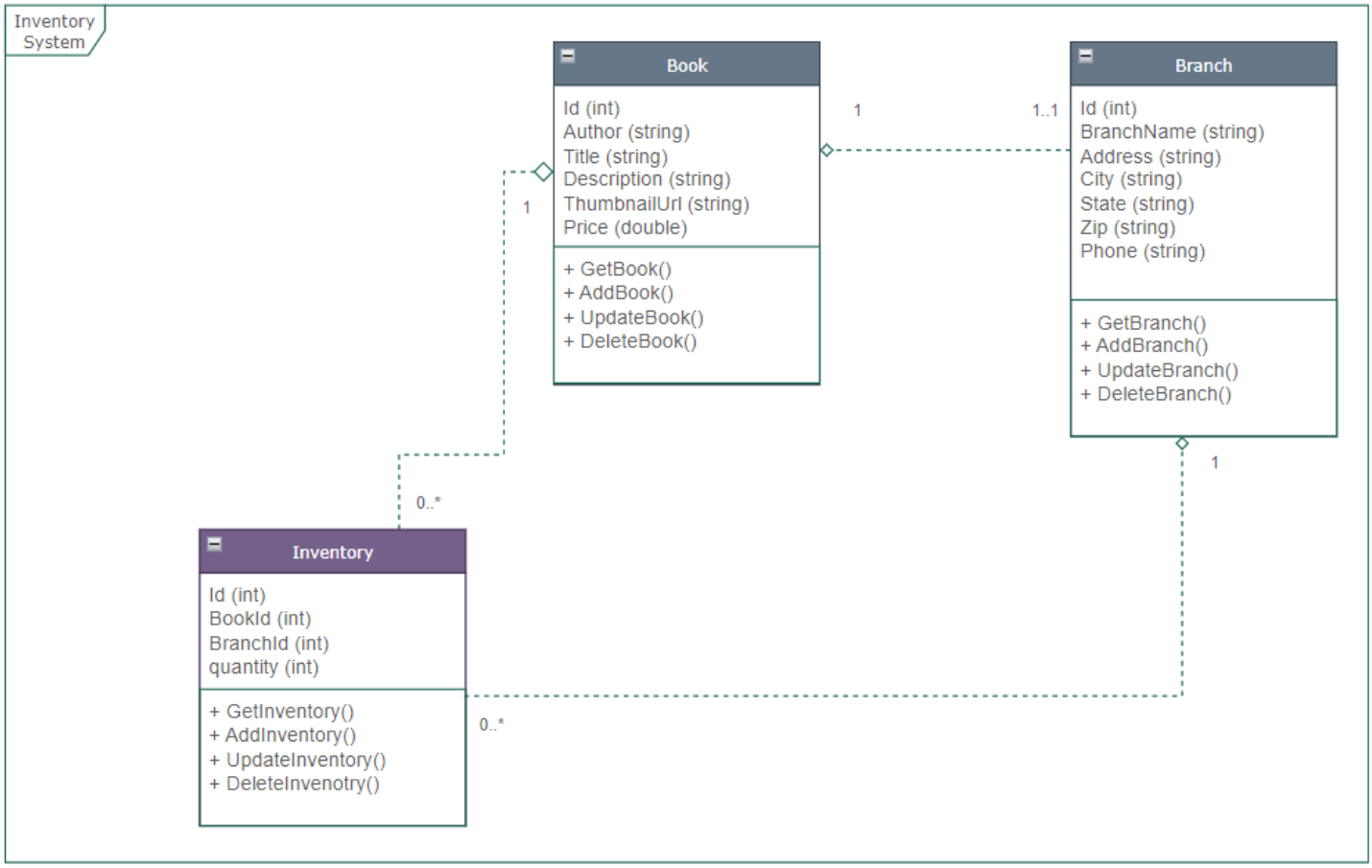
- COMMUNICATION DIAGRAM



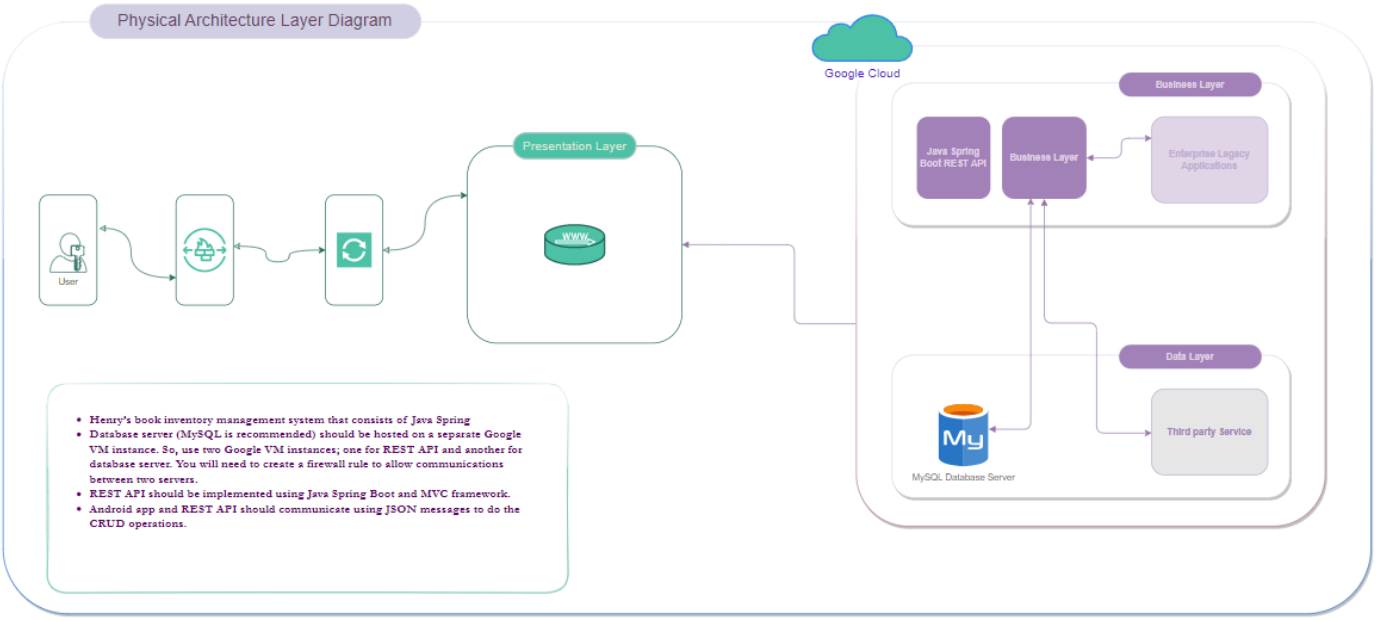
- CRC (CLASS RESPONSIBILITY COLLABORATIONS) CARDS



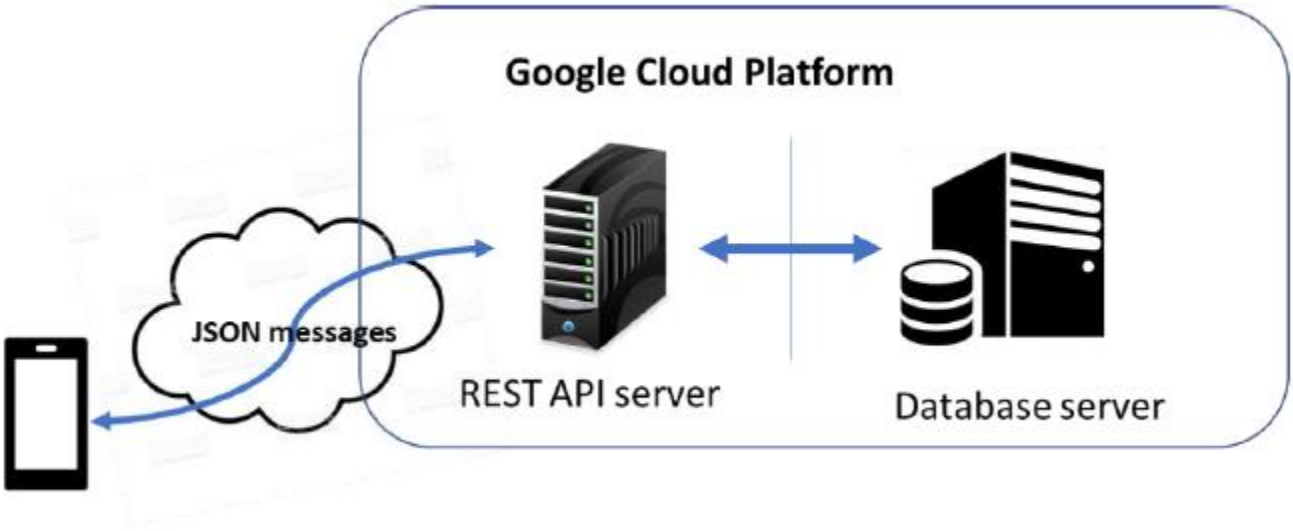
- CLASS DIAGRAM



- PHYSICAL ARCHITECTURAL DIAGRAM



- APPLICATION ARCHITECTURE:

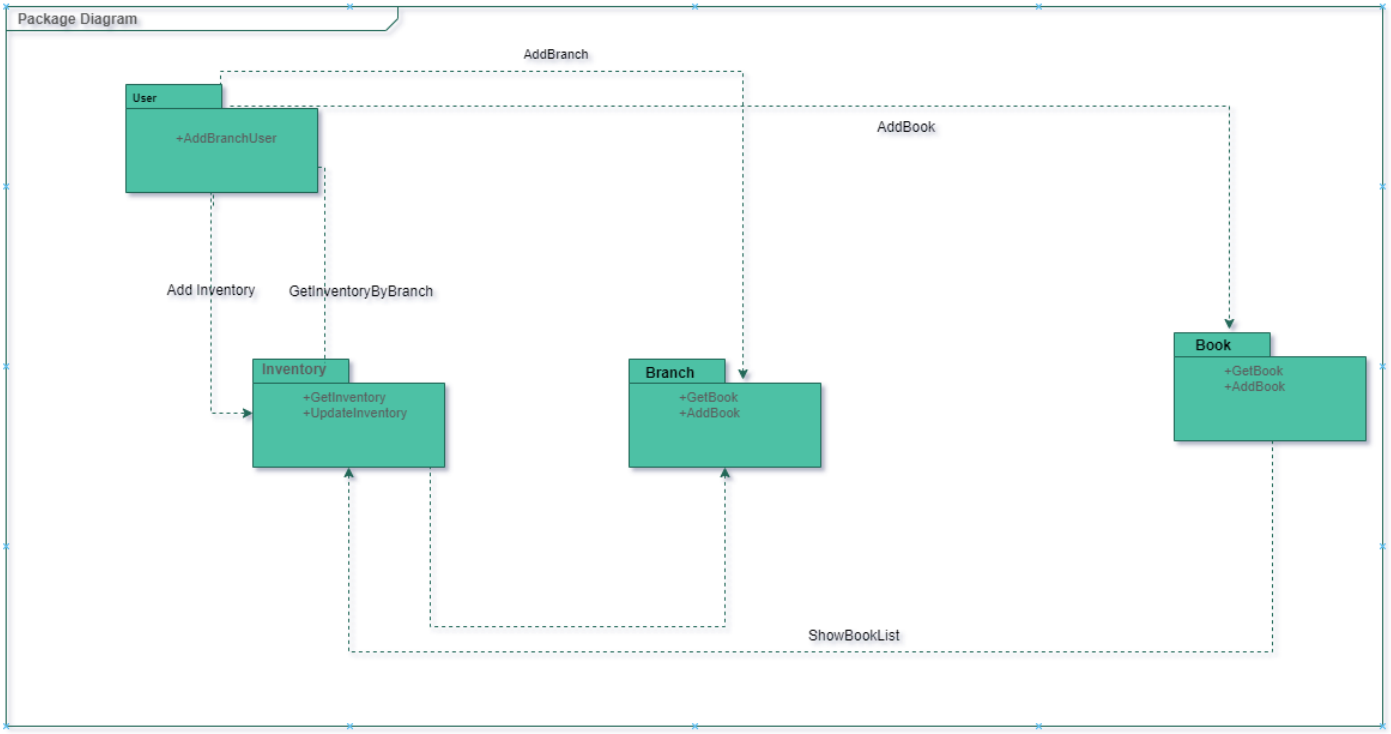


- Ruby on Rails uses the Model-View-Controller (MVC) architectural pattern:

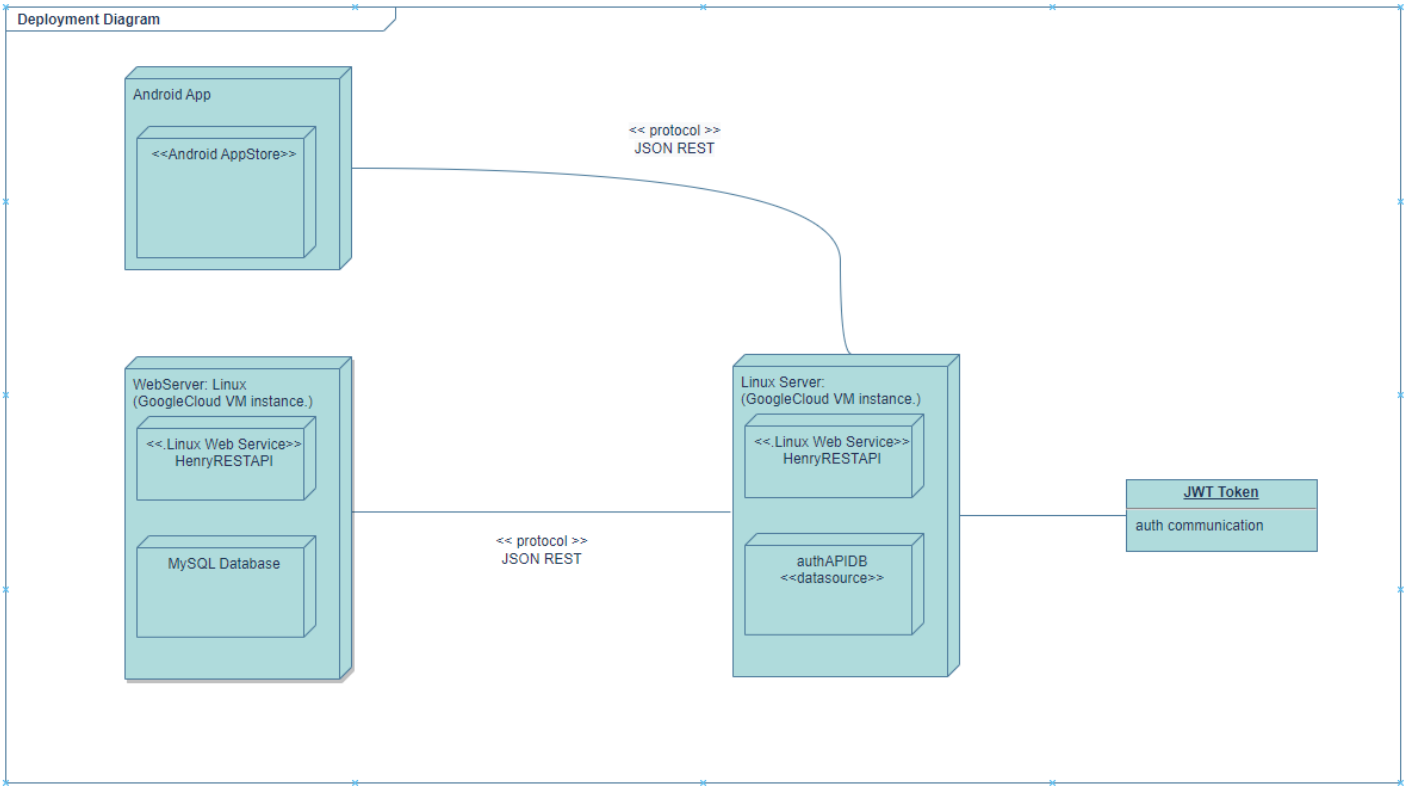
  MVC is a pattern for the architecture of a software application. It separates an application into the following components:

  - Models for handling data and business logic
  - Controllers for handling the user interface and application
  - Views for handling graphical user interface objects and presentation
  - This separation results in user requests being processed as follows:
  - The browser sends a request for a page to the controller on the server
  - The controller retrieves the data it needs from the model in order to respond to the request
  - The controller gives the retrieved data to the view
  - The view is rendered and sent back to the client for the browser to display

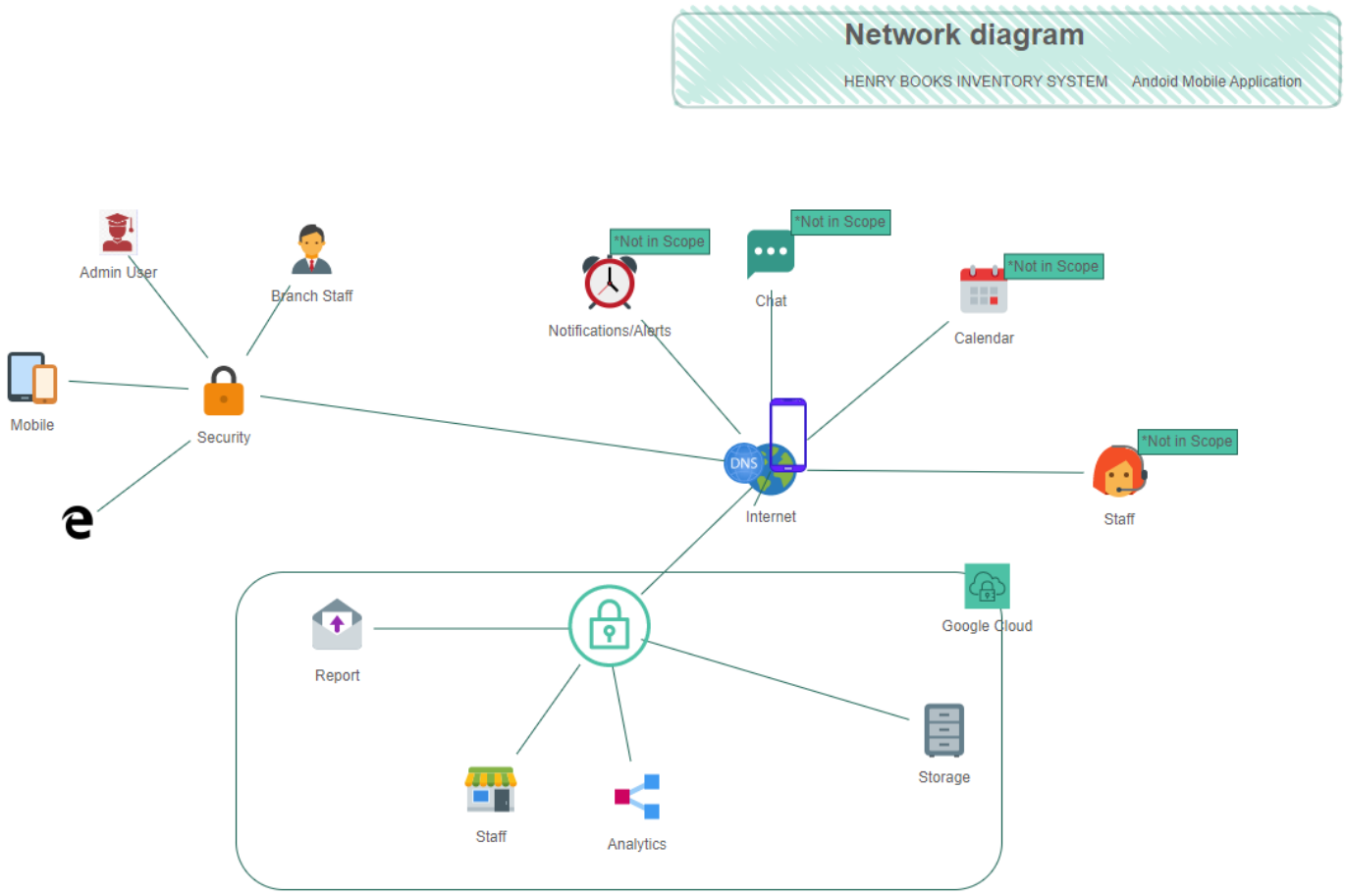
- PACKAGE DIAGRAM



- DEPLOYMENT DIAGRAM



- NETWORK DIAGRAM



• SIZE AND PERFORMANCE

The chosen software architecture supports the key sizing and timing requirements, as stipulated in the Supplementary Specification [7.6]:

1. The system shall support up to 2000 simultaneous users against the central database at any given time, and up to 500 simultaneous users against the local servers at any one time.
2. The system shall provide access to the legacy course catalog database with no more than a 10 second latency.
3. The system must be able to complete 80% of all transactions within 2 minutes.
4. The client portion shall require less than 20 MB disk space and 32 MB RAM.

The selected architecture supports the sizing and timing requirements through the implementation of a client-server architecture. The client portion is implemented on local campus PCs or remote dial up PCs. The components have been designed to ensure that minimal disk and memory requirements are needed on the PC client portion.

• QUALITY

The software architecture supports the quality requirements, as stipulated in the Supplementary Specification [7.6]:

The App user-interface shall be Android Mobile application compliant.

User must have android phone in order access this application

Android phone should have access to the internet in order to access this application

The user interface of the HENRY BOOKS INVENTORY SYSTEM shall be designed for ease-of-use and shall be appropriate for a computer-literate user community with no additional training on the System.

Each feature of the HENRY BOOKS INVENTORY SYSTEM System shall have built-in online help for the user. Online Help shall include step by step instructions on using the System. Online Help shall include definitions for terms and acronyms.

The HENRY BOOKS INVENTORY SYSTEM shall be available 24 hours a day, 7 days a week. There shall be no more than 10% down time.

Mean Time Between Failures shall exceed 300 hours.

- SYSTEMS INVOLVED

Database	MySQL
Programming Language	JAVA
Framework	Spring boot JPA CurdRepository Hibernate Spring Data JPA Spring ORMs
IDE	Android Studio Spring Tool Suite 4
Cloud deployment Platform	Google Cloud
Front end/UI Design	HTML, CSS

DB Connection : spring.datasource.url=jdbc:mysql://10.138.0.5:3306/bookstore

API URL : http://34.145.1.227:8080/api/

DESIGNED BY

Sargam, Ganesh S.

March 04th, 2023