HACETTEPE UNIVERSITY COMPUTER ENGINEERING DEPARMANT
BBM 487 – SOFTWARE ENGINEERING
LABORATORY

LIBRARY BOOK LOAN SYSTEM
- Architecture Notebook-

Group 10
Berat Gökser Atlı-21327656
Hülya Şermin Karakaş-21591198
Sefa Feruzoğlu-21228338

Library Book Loan System
Architecture Notebook

1.   Purpose

This document is designed to illustrate the philosophy, decisions, constraints, justifications, important elements and other comprehensive aspects of the system that are shaped by the design and implementation. Each item will be explained in more detail below. In accordance with the decisions we have made here, the architecture of the library system will be formed.

2.   Architectural goals and philosophy

We aimed at least error and high performance while creating the architecture. So a system that could be adapting to the old system is the first thing we want. To survive the problem of switching from the old system to this system. So our system is web-based and data is being downloaded through the database. The first of the goals is to

fulfill the questionnaire as soon as possible. Designing the database to reduce the cost of queries. Secondly, the changes made in the database are changing rapidly. The system needs to be up-to-date. When someone takes a book or gives it, the user should be able to see it on the system immediately.

3.   Assumptions and dependencies

The current system was largely effective when we took our architectural decisions. When we were doing architectural design, we considered our capabilities and the current system. As a team, we think that we will succeed because we have already done something on the database. Our first target is a good design. The second target is that the updates are at the same time without interrupting the queries. The system can keep up-to-date. There was a lot of resources in this issue and it was effective in our water.

4.   Architecturally significant requirements

5.   Decisions, constraints, and justifications

It allows one call at the same time. For this reason, the results do not interfere. She does not let you take a book that does not exist. If it does, it creates a fault in the database. If there is no book searched, the system may suggest a different book. For the user's help. The date of receipt of the book may be extended. Updated in the database and is not penalized. The book will be followed by who you are. It is known on this occasion when the book will return to the library. If another user wants to buy a book, he can sort out the book by looking at its history. Or it can come and go on that date. This number preserves the number of books in the library. The person who receives the book must return from it. Or punishment.

6.   Architectural Mechanisms

Architectural Mechanism 1
Each user will have a user name and password to get the book and give it back. In this way, it will be possible to follow who the book is.
Architectural Mechanism 2
Is there a book in the library? To be able to determine whether the book user can take it.
Architectural Mechanism 3
Check the time of delivery of the book. To check if books are being delivered on time. The reason for this is to provide control of the books in the library.
Architectural Mechanism 4
The length of the book can be extended. In this case, there is no penalty for not having a book, and the future date of the book is updated.

Architectural Mechanism 5

People who take books are followed. In this way, the number of books in the library will not decrease. If the book does not come, the penalty will be applied.

7.   Key abstractions

8.   Layers or architectural framework

We've done a few things to make the architecture we're building consistent. First we will keep track of the books in the system. According to the book status, the user's purchase status is set. Since the second will be entered with the user name, the user will be able to keep track of which user has which book.

9. Architectural views

Recommended views

- **Logical:** Describes the structure and behavior of architecturally significant portions of the system. This might include the package structure, critical interfaces, important classes and subsystems, and the relationships between these elements. It also includes physical and logical views of persistent data, if persistence will be built into the system. This is a documented subset of the design.
- **Operational:** Describes the physical nodes of the system and the processes, threads, and components that run on those physical nodes. This view isn't necessary if the system runs in a single process and thread.
- **Use case:** A list or diagram of the use cases that contain architecturally significant requirements.