<Project Name>

Architecture Notebook

There is guidance within this template that appears in a style named InfoBlue. This style has a hidden font attribute that allows you to toggle whether it is visible or hidden in this template. Use the Microsoft® Word® menu **Tools > Options > View > Hidden Text** check box to toggle this setting. There is also an option for printing: **Tools > Options > Print**.

# Purpose

Purpose of Architecture Notebook is collecting all properties about system design and standards in one file. These properties include architecture goals, philosophy, assumptions and dependencies, the requirement of design, architectural mechanism, design constraints, justifications, and decisions. At this point, we can easily check whether we follow the rules or not the project after each step. Besides, we can have a standard on the Architectural side of the project. So, we can easily say this document concretize the properties of the system and it helps to the new team member for adapting to the project.

# Architectural goals and philosophy

In this project, we aim to design sports center’s mobile and web application. At this point, our project is divided into two different sides. These are member and manager sides. In member side, we can have a wide variety of user profiles. So, the first architectural goal is simplicity. It is important to us that users can use the application easily. The members may want to connect with many different devices. Therefore, we need to provide various device support. (For this title, we decide to use Android version 4 or newer). In the mobile application, we aimed to keep the size of the mobile application and system requirements as less as possible. So, we can reach maximum population of members. In management side, we need to share statistics of centers clearly with managers. A misunderstanding at this topic can cause bad sequences for the firm. Our system doesn’t need any additional hardware specializations. In website, we work on modern web browser. But, we plan to increase browser support.

# Assumptions and dependencies

In this project, we will develop web and android application for sports center management. We use Angular5 for web application side because of javascript languages are easier than other languages which are using for web application’s implementation. Also, our team members have no knowledge about web designing and implementing so, we have to learn fast and pass on implementing.

We use and android because of implementing an android application’s implementation is easier than other mobile platform applications, android is widely used and our team members have knowledge about it.

We use MySQL for database side, because of easier and simpler than other database services and our team have knowledge about it. Also, our server supports MySQL and server processes about database became easier.

We use java for control database, web and android applications’ base. Firstly, we have to use and object-oriented language like c# or java. Secondly, we are more experienced about java programming. Also, java community is better than c# community, any problem’s solution is easier to find for java.

# Architecturally significant requirements

### Why Use the Spring Framework?

<http://www.wrox.com/WileyCDA/Section/Why-Use-the-Spring-Framework-.id-130098.html>

### Why JavaScript Is and Will Continue to Be the First Choice of Programmers?

<https://dzone.com/articles/why-javascript-and-will>

### Why is the Android OS So Popular?

<http://opensourceforu.com/2016/02/why-is-the-android-os-so-popular/>

### Why MySQL is still king?

<https://www.infoworld.com/article/3195764/nosql/nosql-no-problem-why-mysql-is-still-king.html>

[Insert a reference or link to the requirements that must be implemented to realize the architecture.]

NEDEN OOP KULLANDIK NEDEN SPRİNG KULLANDIK MAKALEYLE AÇIKLA

# Decisions, constraints, and justifications xxxx

[List the decisions that have been made regarding architectural approaches and the constraints being placed on the way that the developers build the system. These will serve as guidelines for defining architecturally significant parts of the system. Justify each decision or constraint so that developers understand the importance of building the system according to the context created by those decisions and constraints. This may include a list of DOs and DON’Ts to guide the developers in building the system.]

* The system should be developed in accordance with Object Oriented Design. OO design is important for future development and it is more useful for teams.
* The system’s official language is Java. Besides that, we should use Angular 2+ in frontend side. In mobile application, we should use Java for Android. Java has great support in server side and it required for Android mobile application development. Angular 2+ is also use class structure, so it is most available front-end framework for us.
* Model-View- Controller is the system’s architectural pattern. In this way, we can separate critical system parts easily. It decreases cost of maintenance.
* The system’s database management system is MySQL. Because it is open source and we can easily learn how to use it.
* In mobile side, the application supports 4.2 or newer versions of Android. It provides wide support for maximum users.
* In website, the website supports Google Chrome, Firefox, Opera and Microsoft Edge. It provides wide support for maximum users.
* The Server is available for every time. So, we need to use cloud servers. In this case, we rent a server from Digital Ocean.
* In front end development, we use bootstrap 4 for better and modern view.
* The development environments; Visual Studio Code for Web development, Android Studio for Mobile application, and Sts tool for back-end development. It is important for team works. Because it decreases inconsistency.
* Web application runs on apache server. Its open source software and it has large support.

# Architectural Mechanisms

[List the architectural mechanisms and describe the current state of each one. Initially, each mechanism may be only name and a brief description. They will evolve until the mechanism is a collaboration or pattern that can be directly applied to some aspect of the design.]

## Availability

The system will be active whenever the user wishes to access it on mobile and web application.

## Archiving

The user's past training information will be stored in the database.

## Graphics

User interfaces will be as simple as possible for the user’s easy use.

## Communication

The user will be able to make calls at any time using the contact information.

## Event Management

System will show events to the user according to user situation.

## Security

The user information will be kept encrypted in the database.

## Data Management

All data related to the application will be kept using the mysql database. Mobile and web application will use the data in the database via web service.

## Memory Management

Some personal data will be kept in local memory on the user's device.

## Mail

The user will be notified via e-mail when necessary.

# Key abstractions

[List and briefly describe the key abstractions of the system. This should be a relatively short list of the critical concepts that define the system. The key abstractions will usually translate to the initial analysis classes and important patterns.]

Users:

* Visitor: Whoever created an account but has not paid money,
* Member: Whoever created an account and paid money
* Trainer: Whoever is responsible for coaching members and planning their activity.
* Manager: Whoever is responsible for the management of the sports center
* Owner: Whoever owns the gym

Interfaces: It creates connection between users and system.

Models: It consists of system’s main components.

* Courses
* Centers
* All user types

Database: It holds all information about the sports center and user

# Layers or architectural framework

[Describe the architectural pattern that you will use or how the architecture will be consistent and uniform. This could be a simple reference to an existing or well-known architectural pattern, such as the Layer framework, a reference to a high-level model of the framework, or a description of how the major system components should be put together.]

# The system consists of 3 different layers. The first layer is Presentation layer. This layer is responsible from the view of the project. We develop both mobile and web application. So, we have two different interfaces. These interfaces include many view classes. The second layer is the business layer. This layer is responsible from creating communication between the presentation layer and data layer. The business layer gets user’s demand and provides them. The last layer is data layer. The data layer provides resources to the system. This layer includes database connection. The business layer reaches necessary information with the data layer. This layer is also responsible from the connection between other systems.

# Architectural views

[Describe the architectural views that you will use to describe the software architecture. This illustrates the different perspectives that you will make available to review and to document architectural decisions.]

By using 4+1 architectural view model, we got logical, development, process and physical 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.

**BURAYA CLASS DIAGRAM GELECEK**

* **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.
* **Operational process view olabilir ?**
* **Use case:** A list or diagram of the use cases that contain architecturally significant requirements.