JetLAG

Student Organization Software
Software Architecture Document

Version <1.0>

[Note: The following template is provided for use with the Unified Process for EDUcation. Text enclosed in square brackets and displayed in blue italics (style=InfoBlue) is included to provide guidance to the author and should be deleted before publishing the document. A paragraph entered following this style will automatically be set to normal (style=Body Text).]

[To customize automatic fields in Microsoft Word (which display a gray background when selected), select File>Properties and replace the Title, Subject and Company fields with the appropriate information for this document. After closing the dialog, automatic fields may be updated throughout the document by selecting Edit>Select All (or Ctrl-A) and pressing F9, or simply click on the field and press F9. This must be done separately for Headers and Footers. Alt-F9 will toggle between displaying the field names and the field contents. See Word help for more information on working with fields.]

Revision History

Date	Version	Description	Author
10/23/2022	1.0	First iteration of document.	JetLAG Team

Table of Contents

Software Architecture Document

1. Introduction

1.1 Purpose

This Software Architecture Document covers the architectural decisions that have been made in creating the Student Organizational Software.

1.2 Scope

This document provides insight into the architectural design of the Student Organizational Software, including the Goals and Constraints which guided development. Other documents are referenced to provide further information.

1.3 Definitions, Acronyms, and Abbreviations

See the glossary provided in the Supplementary Requirements Specifications document.

1.4 References

1. SOS Supplementary Specifications Document – Glossary
2. SOS Use Case Realization Document
3. SOS Software Requirements Specifications
4. … (Adding as Referenced)

1.5 Overview

Sections 2 through 8 cover the various views of the design of SOS, goals and constraints, as well as performance and quality goals.

2. Architectural Representation

The views and models presented in this document as well as the Use Case Realization document are represented in Unified Modeling Language.

3. Architectural Goals and Constraints

SOS is designed to be an online software solution users can access in browser and on mobile devices. SOS is composed of three main components: a Flutter front-end app, a data-broker, and a back-end Cosmos DB database to store information.

4. Use-Case View

The Use-Case View models the scenarios and use cases that are the focus in development. They show what scenarios and use cases have been taken under consideration in the architectural design of SOS.

4.1 Use-Case Realizations

Refer to Use Case Realizations document.

5. Logical View

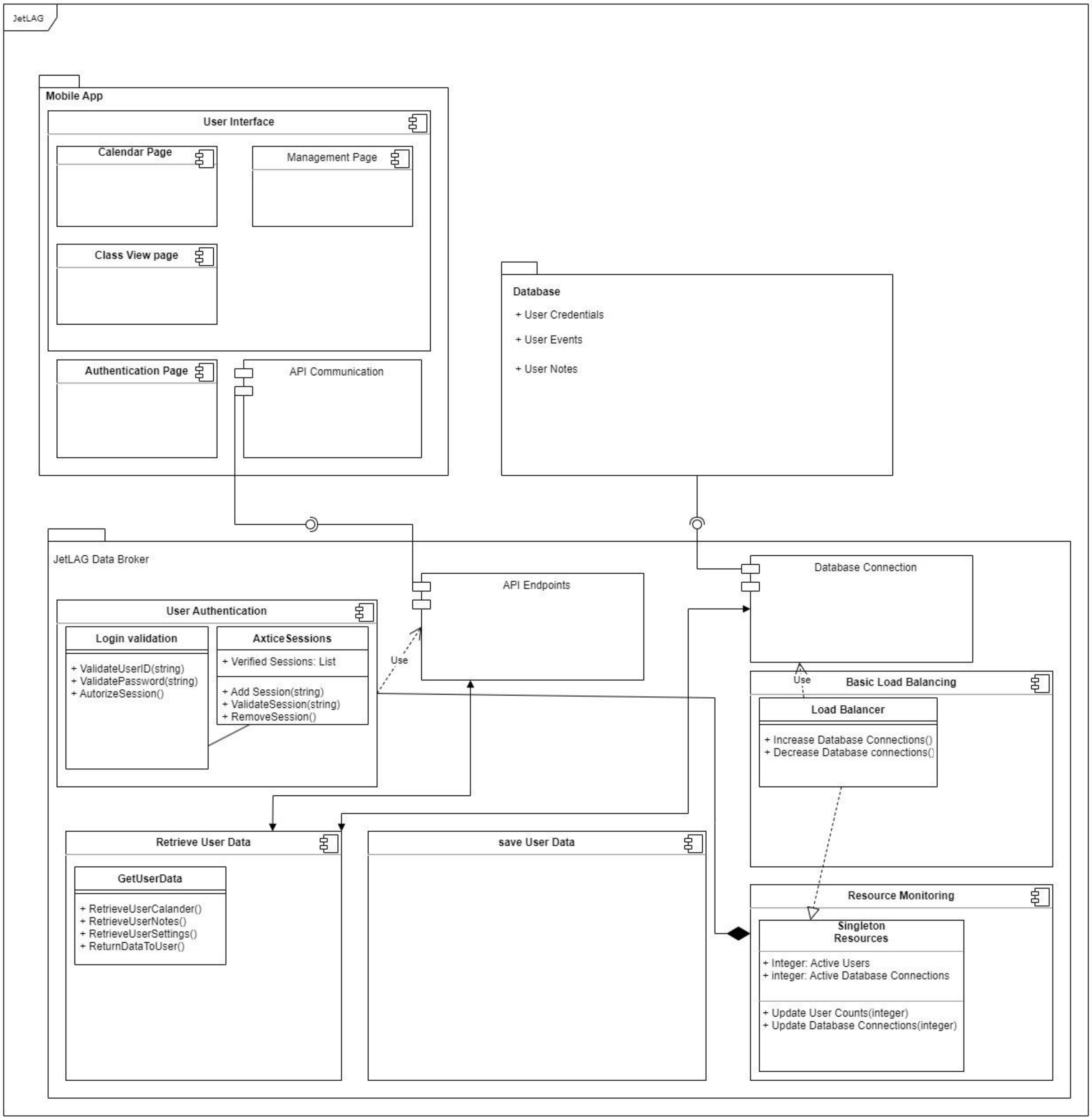
[This section describes the architecturally significant parts of the design model, such as its decomposition into subsystems and packages. And for each significant package, its decomposition into classes and class utilities. You should introduce architecturally significant classes and describe their responsibilities, as well

as a few very important relationships, operations, and attributes.]

5.1 Overview

In our diagram you can see we have 3 packages representing the 3 components of our system, the mobile app, data Broker, and database. In general the mobile app makes api requests to the data broker, from there the data broker determines what data I needs to save and retrieve from the database.

5.2 Architecturally Significant Design Packages



6. Interface Description

the Mobile app and data broker communicate over API requests, in turn the Data broker communicates to the database over JDBC connections.
[A description of the major entity interfaces, including screen formats, valid inputs, and resulting outputs. If a User-Interface Prototype Document is available, refer to it in this section]

7. Size and Performance

SOS architecture is designed to meet size and performance goals set in the Software Requirements Specifications and the Supplementary Specifications Requirements. Those requirements were identified to offer quick performance while minimizing disk storage size.

8. Quality

SOS architecture is designed to meet quality standards set forth in the Software Requirements Specifications and the Supplementary Specifications Requirements.