SOFTWARE DESIGN DOCUMENT

*for*

Tutor Reservation Application

By

Priscilla Udomprasert

3/8/2025

1. Introduction

This software design document is an overview architecture blueprint for a tutoring reservation application for Team-Boom, Inc.

* 1. . Purpose

The purpose of this document is to outline the design for a tutoring reservation application that allows smartphone customers to search available tutors and schedule tutoring sessions. It also allows for tutors to manage and edit their availabilities. This mobile application will function similarly to a hotel reservation app but will be tailored for parents to book tutors. Parents will be able to access each tutor’s profile where they will be able to view their educational background, performance rating, and pricing.

1.2. Scope

This application will provide an intuitive interface for users. It will include features such as user authentication, payment processing, and administrative management.

1.3. Definitions, Acronyms, and Abbreviations

* User – An individual using this application to search for and schedule tutors.
* Tutors – An individual who manages their professional services such as availability editing.
* Admin – An individual with special privileges who manages the entire system.
* Payment System – A third party service to process payments in a secure manner.

1. ARCHITECTURAL DESIGN

The application consists of a frontend, backend, and database server. The frontend is responsible for user interactions and the backend handles business logic while the database server stores the data.

2.1. Overview Architecture

* Frontend: React Native for smartphone application usage
* Backend: Node js with Express.js
* Database server: MySQL for structuring and storing data.
* Authentication: OAUTH 2.0 and JWT for secure access
* Payment System: N/A

2.2. Deployment

* Cloud Provider AWS
* Code Depository: Git for version control system

1. FUNCTION REQUIREMENTS

* Customers
  + - \*\*Sign Up/Login\*\* (Email, Google, Facebook, Apple ID)
  + - \*\*Search & Filter Teachers\*\* (By subject, rating, location, price, online/onsite availability)
  + - \*\*View Teacher Profiles\*\* (Photo, bio, experience, education, ratings, reviews, pricing, availability)
  + - \*\*Booking System\*\* (Select a time slot, online or onsite, payment)
  + - \*\*Chat with Teachers\*\* (Pre-booking chat for inquiries)
  + - \*\*Manage Bookings\*\* (View upcoming/past bookings, reschedule/cancel)
  + - \*\*Payment System\*\* (Credit/Debit card, e-wallets, bank transfer)
  + - \*\*Leave Reviews & Ratings\*\* (After a session)
* Tutors
  + - \*\*Sign Up/Login\*\* (Email, Google, Facebook, Apple ID)
  + - \*\*Manage Availability\*\* (Add and update their own available time slots)
  + - \*\*Create & Manage Profile\*\* (Subjects, pricing, credentials, upload certificates)
  + - \*\*Manage Bookings\*\* (Accept/decline bookings, reschedule)
  + - \*\*Messaging System\*\* (Chat with parents before and after bookings)
  + - \*\*Earnings Dashboard\*\* (Track earnings, request payouts, see payment history)
  + - \*\*Reputation & Reviews\*\* (See ratings, respond to reviews)
* Admin
  + - \*\*Dashboard\*\* (Manage users, bookings, payments, disputes)
  + - \*\*Add & Approve Teachers\*\* (Create new teacher profiles and verify credentials before listing them)
  + - \*\*Manage Reviews & Complaints\*\*
  + - \*\*Generate Reports\*\* (Revenue, bookings, teacher performance)

1. NON-FUNCTIONAL REQUIREMENTS

* Scalability: The system should handle a high number of users and transactions.
* Security: Secure authentication and encrypted transactions.
* Performance: Optimized response times for searches and bookings.

1. SERVER ENDPOINT

* Authentication
  + POST/API/register
  + POST/API/tutors/:id
* Tutors (view Profile)
  + Get/API/tutors
  + Get/API/tutors/:id
* Bookings
  + POST/API/booking
  + Get/API/booking/:id
  + POST/API/booking/reschedule/:id
  + POST/API/booking/cancel/:id
* Tutor\_Schedules
  + Get/API/tutors-schedules/:id
* Tutor\_Review
  + POST/API/tutor\_review/:id
  + Get/API/tutor\_review/:id
* Payment
  + POST/API/payment/:id
  + Get/API/bankpayment/:id
* Tutor\_Manage
  + Get/API/tutor\_manage/profile/:id
  + POST/API/tutor\_manage/profile/:id
  + Get/API/tutor\_manage/booking/:id
  + POST/API/tutor\_manage/booking/:id
  + Get/API/tutor\_manage/dashboard/:id
* Admin
  + Get/API/admin/user/:id
  + POST/API/admin/user/:id
  + Get/API/booking/user/:id
  + POST/API/booking/user/:id
  + Get/API/payment/user/:id
  + POST/API/payment/user/:id
  + Get/API/admin/review/:id
  + Post/API/admin/review/:id
  + Get/API/admin/report/:id

1. DATABASE DESIGN