Team 9

WiFind - A Platform for Connectivity

Software Design Document

Walid Abdullahi, Kelsi Hill, Kevin Morales Funes,
Lisa Nguyen, Thomas Sigler

Table of Contents

1.0: Introduction

1.1 - Purpose: This software engineering design document describes the architecture and system design of the website WiFind. It is intended to be read by software developers and is written with this audience in mind.

1.2 - Scope: WiFind is a ecommerce website focused on providing a platform for sellers to monetize WiFi and provide consumers with temporary access to public, secure WiFi. The website is designed with ease of use and safety in mind, requiring significant information from each user and each listing for security. The goal of this software is to provide and easy and secure method of accessing WiFi on the go for the average consumer.

1.3 - Overview: This document goes over the design of the system and the decisions and philosophy that ultimately led to these design decisions. Included is diagrams providing an overview of the system from various perspectives. Firstly, there will be an overview of the system and its architecture. What follows will be the data design and the components that work with the data. Finally, this document will have the human interface design and requirements matrix.

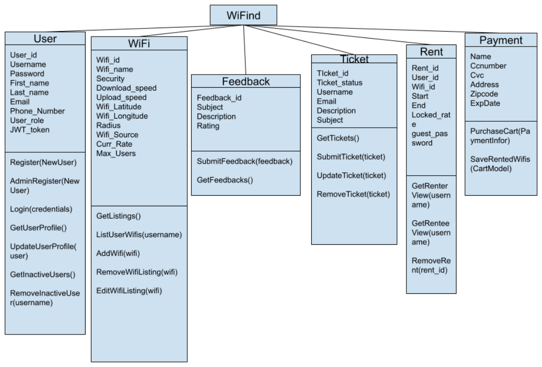
2.0: System Overview

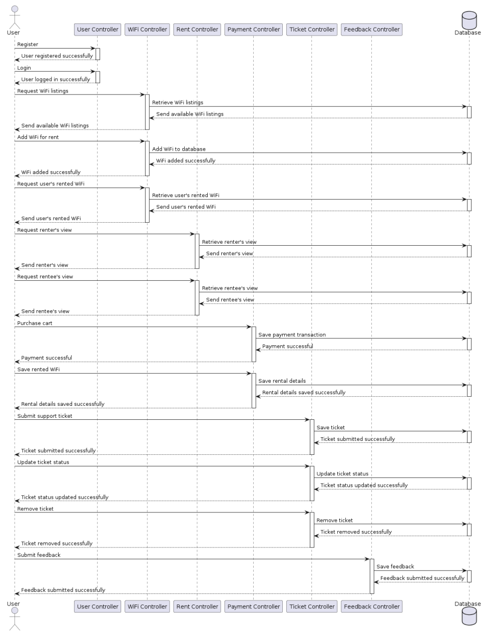
WiFind is split into 5 modules all connected by a main page. These five modules are a payment system, a user management system, a notification system, a ticketing system, and a WiFi management system. This design provides an interactive and easy experience that allows safe and reliable internet access on the go. The design consists of tabs connected to the main page that the user can click on to go to pages handling each module, such as listings and profile management. They are clearly bifurcated, but work well together to provide a seamless and cohesive experience.

3.0: Architecture Overview

3.1 - Architectural Design: The program consists of the five aforementioned modules. These modules are largely independent in their implementation, but rely on core data structures of user ID and WiFi ID. User ID is held central in the User data and are used throughout the feedback system and ticket systems as unique identifiers. WiFi ID is a unique identifier for each listed WiFi and is used to associate listed WiFis with both owner users and renting users, being added to the user profile. This data is the primary object that connects the independent modules together in a coherent system. This allows the user to seamlessly go from managing their profile to managing their WiFi listings and rentals while providing peripheral functionality such as tickets and feedback for system improvement.

3.2 - Decomposition Description: The system as designed is firmly rooted in object-oriented programming both for the frontend and the backend of all the modules. There is little reliance of each module on the others, allowing independent functionality as needed. One module failing won't cause the others to fail. This is done by allowing the modules to exchange data between each other and the central server and have some interaction, but with each keeping its functionality almost entirely "in-house". This object-oriented design is done using the MVC methodology of .NET, allowing the controller for each module a great deal of independence as long as the data that it receives is properly organized when the API is called.





3.3 - Design Rationale: This modular design was chosen for ease of organization and user of the project. One significant drawback that would hold the project back from further development is difficulty in integrating the modules directly with each other. Modularness means that everything must be processed through the database for the server, delaying data transfers and making the website slightly slower. We also considered using more interconnected models centered on data flow, but considered this too complex to build and requiring complex debugging that would hinder quick development.

Data Design

4.1 - Data Description: WiFind's data is stored in a SQL database on many different tables. A few of them are the user, WiFi, Tickets, and Feedback. Additionally transactions are stored with User ID's and WiFi ID's, which act as key pieces of data for unique identification. Data is processed through control models that require data be organized in a certain way for processing, denying processing otherwise.

4.2 - Data Dictionary:

Feedback Controller - Database, SubmitFeedback(feedbackregmodel), GetAllFeedbacks()

Payment Controller - Database, PurchaseCart(paymentmodel), SaveRentedWifis(cartmodel)

Rent Controller - Database, GetRenteeView(username), GetRenterView(username), RemoveRent(Rent_Id)

Ticket Controller - Database, SubmitTicket(supportticketregmodel), UpdateTicket(updateticketmodel), RemoveTicket(ticketmodel)

User Controller - Database, Register(newusermodel), AdminRegister(newusermodel), Login(credentials), GetUserProfile(), UpdateUserProfile(updateprofile), GetInactiveUsers(), RemoveInactiveUsers(username)

Wifi Controller - Database, GetListings(), ListUserWifi(username), AddWifi(wifireg), removeWifiListing(wifi), RemoveWifiListing(wifi)

<u>Component Design</u>

User: Register adds a new user to the website database.AdminRegister adds new administrator accounts. Login allows users to log in to their existing accounts. UpdateUserProfile allows the user to change the details of their profile. GetInactiveUsers returns a list of all users who have not logged on within 3 months. RemoveUser removes the given username.

Wifi: GetListings returns all current listings on the website. ListUserWifis lists all wifis currently being rented by a user. AddWifi allows users to list wifis available for rent. RemoveWifiListing delists currently listed wifis. EditWifiListing allows users to alter details about existing wifi listings.

Rent: GetRenterView returns all of the listings currently being rented by a specific user. GetRenteeView gets all the Wifis currently being listed by a user for rent. RemoveRent ends a rental that is happening between two users.

Ticket: GetTickets returns a list of all tickets. SubmitTicket allows users to create new tickets for support. UpdateTicket allows administration to change ticket status. RemoveTicket allows administration to remove a ticket from the database.

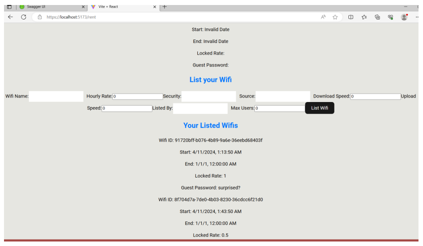
Payment: PurchaseCart is the payment process that allows a user to buy all currently held Wifis in their cart. SaveRentedWifis saves each individual wifi purchased and its details into the database.

Feedback: SubmitFeedback allows users to submit their feedback to the website, which is stored in the database. GetFeedbacks returns a list of all feedbacks on the website.

<u>User Interface Design</u>

<u>6.1 - Overview of User Interface</u>: The user is greeted with a homepage with a bar of buttons at the top that take the user to each of the previously described features. These buttons take the user to pages for each individual modules. These modules interact with a few modules for independence and are focused on providing a singular set of services from each controller. Some of these modules are interconnected, such as the payment module and the rent module, which work in tandem to process payments and store transactions. The UI generally consists of textboxes for inputting information into control models for API calls and buttons for submissions.

<u>6.2 - Screen Images</u>:



Start: Invalid Date

End: Invalid Date

Locked Rate:

Guest Password:

Wifi Name: Hourly Rate: 0 Security: Source: Download Speed: 0 Upload Speed: 0 Listed By: Max Users: 0

Wifi ID: 91720bff-b076-4b89-9a6e-36eebd68403f

Start: 4/11/2024, 1:13:50 AM

End: 1/1/1, 12:00:00 AM

Locked Rate: 1

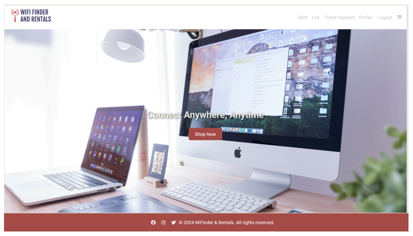
Guest Password: surprised?

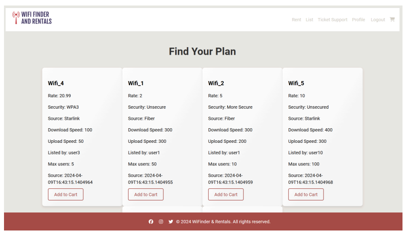
Wifi ID: 8f704d7a-7de0-4b03-8230-36cdcc6f21d0

Start: 4/11/2024, 1:43:50 AM

End: 1/1/1, 12:00:00 AM

Locked Rate: 0.5


WIFI FINDER AND RENTALS
Connect Anywhere, Anytime
Shop Now
© 2024 WiFinder & Rentals. All rights reserved.



Wifi_4	Wifi_1	Wifi_2	Wifi_5
Rate: 20.99	Rate: 2	Rate: 5	Rate: 10
Security: WPA3	Security: Unsecure	Security: More Secure	Security: Unsecured
Source: Starlink	Source: Fiber	Source: Fiber	Source: Starlink
Download Speed: 100	Download Speed: 300	Download Speed: 300	Download Speed: 400
Upload Speed: 50	Upload Speed: 300	Upload Speed: 200	Upload Speed: 300
Listed by: user3	Listed by: user1	Listed by: user1	Listed by: user10
Max users: 5	Max users: 50	Max users: 10	Max users: 100
Source: 2024-04-09T16:43:15.1404964	Source: 2024-04-09T16:43:15.1404955	Source: 2024-04-09T16:43:15.1404959	Source: 2024-04-09T16:43:15.1404968

6.3 - Screen Objects and Actions: The buttons at the top take the user to pages for different controllers that accomplish different tasks on the website. The input fields input data for the control models and buttons submit data to their connected methods along with making an API call.

Requirements Matrix

User Controller	1, 4, 6, 10, 13, 14, 15
WiFi Controller	4, 8, 9
Payment Controller	2, 11
Rent Controller	4, 7
Ticket Controller	3, 4, 5
Feedback Controller	4