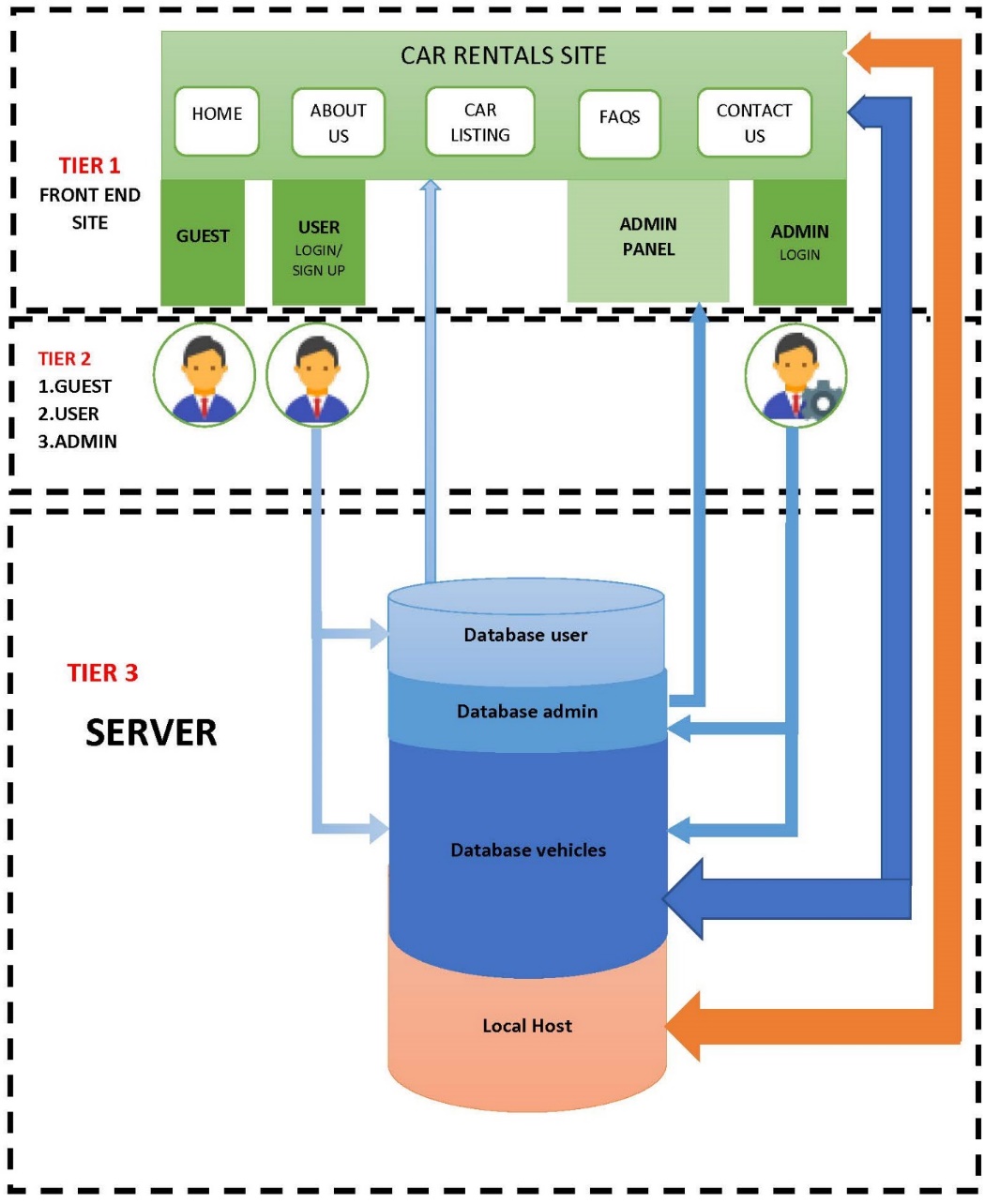
**3. SYSTEM ARCHITECTURE**

**3.1 Architectural Design**



**Relations between**

Tier 1 – Tier 2

a. Guests can only visualize the car rental website without having access to other User-only options such as booking a car, customizing profile and sending testimonials.

b. User

- can login through the interface;

- can visualize the car rental website;

- is able to book a vehicle, customize their profile and send testimonials;

c. Administrator

- can login through the interface using special admin login button;

- can visualize the car rental website;

- can manage many things through the admin panel, such as adding a vehicle, manage bookings, etc.;

Tier 2 – Tier 3

a. Guest has no relation here. It would be possible to implement a special section in the database for newsletter.

b. User

-the Login interface sends a query to the database that’s on the Apache server;

- as the user performs command such as bookings, it gets added to the database;

c. Administrator

- admin login interface sends a query to the database;

- whenever the administrator adds new vehicles or changes information about users, these changes get added to the database;

Tier 1 – Tier 3

* the data introduced by the user and administrator from the server change the website’s appearance;
* the back-end of this website is located in the local host (XAMPP + Apache Server);

**3.2 Decomposition Description**

**3.3 Design Rationale**

The approach for this car rental website architecture has been selected because it is probably the most common; it is usually built around the database, and many applications in business naturally lend themselves to storing information in tables.

Many of the biggest and best software frameworks—like Java EE, Drupal, and Express—were built with this structure in mind, so many of the applications built with them naturally come out in a layered architecture.

The architecture is arranged so the data enters the top layer and works its way down each layer until it reaches the bottom, which is usually a database. Along the way, each layer has a specific task, like checking the data for consistency or reformatting the values to keep them consistent. It is the standard software development approach offered by most of the popular web frameworks and it is a layered architecture. Just above the database is the model layer, which often contains business logic and information about the types of data in the database. At the top is the view layer, which is often CSS, JavaScript, and HTML with dynamic embedded code. In the middle, you have the controller, which has various rules and methods for transforming the data moving between the view and the model.

The advantage of a layered architecture is the separation of concerns, which means that each layer can focus solely on its role. This makes it:

* Maintainable
* Testable
* Easy to assign separate "roles"
* Easy to update and enhance layers separately

Proper layered architectures will have isolated layers that aren’t affected by certain changes in other layers, allowing for easier refactoring. This architecture can also contain additional open layers, like a service layer, that can be used to access shared services only in the business layer but also get bypassed for speed.

**Best for:**

* New applications that need to be built quickly
* Enterprise or business applications that need to mirror traditional IT departments and processes
* Teams with inexperienced developers who don’t understand other architectures yet
* Applications requiring strict maintainability and testability standards

**7. REQUIREMENTS MATRIX**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **REQUIREMENTS**  **TRACEABILITY MATRIX** | | | | | | |
| Project Name: Car Rental Website | | | | | | |
| User class ID# | User Class Name | Functional Requirement ID# | Functional Requirement Name/Use Case | System Component | Priority | Test Case ID# |
| UC\_1 | Customer User | 1.1 | Website access | Link | High | #TC001 |
|  |  | 1.2 | Update notification | Newsletter | Medium | #TC002 |
|  |  | 1.3 | User Registration | Login/Register | High | #TC003 |
|  |  | 1.4 | User Login | Login/Register | High | #TC004 |
|  |  | 1.5 | Password retrieval | Login/Register | High | #TC005 |
|  |  | 1.6 | Rental search | Search module | High | #TC006 |
|  |  | 1.7 | Rental selection & booking | Booking module | High | #TC007 |
|  |  | 1.8 | Profile update | Profile module | High | #TC008 |
|  |  | 1.9 | Check bookings | Profile module | High | #TC009 |
|  |  | 1.10 | Post testimonial | Profile module | High | #TC010 |
|  |  | 1.11 | Sign out and Log back in | Login/Register | High | #TC011 |
| UC\_2 | Administrator User | 2.1 | Admin Login | Super Login module | High | #TC012 |
|  |  | 2.2 | Create/Delete Vehicle listing | Admin Panel  module | High | #TC013 |
|  |  | 2.3 | Post Vehicle listing | Admin Panel  module | High | #TC014 |
|  |  | 2.4 | Confirm/Cancel customer booking | Admin Panel  module | High | #TC015 |
|  |  | 2.5 | Manage testimonial | Admin Panel  module | High | #TC016 |
|  |  | 2.6 | Contact us query management | Admin Panel  module | High | #TC017 |
|  |  | 2.7 | Check User Details | Admin Panel  module | High | #TC018 |
|  |  | 2.8 | Update public details | Admin Panel  module | High | #TC019 |
|  |  | 2.9 | Manage newsletter subscribers | Admin Panel  module | High | #TC020 |
|  |  | 2.10 | Dashboard menu statistics | Admin Panel  module | High | #TC021 |
|  |  | 2.11 | Change admin password | Admin Panel  module | High | #TC022 |
|  |  | 2.12 | Log out of Admin Panel | Admin Panel  module | High | #TC023 |
| UC\_3 | Guest User | 3.1 | Explore website | Link | High | #TC024 |
|  |  | 3.2 | Booking restriction | Login/Register  prompt | High | #TC025 |
|  |  | 3.3 | Contact admin | Contact us page | High | #TC026 |
|  |  | 3.4 | Transition into Customer User | Login/Register | High | #TC027 |
|  |  | 3.5 | No interaction with admin, booking-wise | Login/Register  prompt | High | #TC028 |
|  |  | 3.6 | Update notification | Newsletter | Medium | #TC029 |
|  |  | 3.7 | Encounter illegal action | Login/Register prompt | High | #TC030 |