Wander Wave Tourism

Muhammed Saad
Masters of Software Engineering for Industrial Application
Garduate School, Hof University of Applied Sciences
Hof, Bavaria, Germany
muhammad.hassan@hof-university.de

Aziz Rahman Atal
Masters of Software Engineering for Industrial Application
Garduate School, Hof University of Applied Sciences
Hof, Bavaria, Germany
aziz.atal@hof-university.de

Arslan Zafar
Masters of Software Engineering for Industrial Application
Garduate School, Hof University of Applied Sciences
Hof, Bavaria, Germany
arslan.zafar@hof-university.de

***Abstract*— A comprehensive solution, based on the Yii2 basic and advanced frameworks, shall be presented in this report to manage tour packages, purchase tour packages or make payment of tour packages. In addition, it is discussed how to integrate ownCloud into the efficient sharing of files. Increasing demand for complete tour package management, and the need to process payments safely and efficiently, is at the heart of this work. The lack of a common system combining package management, reservation, payment processing and file sharing functionalities is addressed. In order to address this problem, we have proposed a methodology that uses the Yii2 frameworks' ability to create high performance and scalable applications. In order to achieve effective file sharing and collaboration, the system provides a number of features including package creation, booking management, secure online payment channels as well as seamless integration with ownCloud. The results show improvements in efficiency, enhanced safety and simplification of procedures related to package management, booking, payment and file sharing. The overall aim of this solution is to improve the operation of tour agencies and clients in general by providing a comprehensive and coherent platform.**

Keywords*— *tourism application, tours available, tour details, booking tour packages, tour payments, tour plan visitor, booking management, media files uploading, media files sharing

I. INTRODUCTION

This paper is all about the Wander Wave Tourism (WWT) Application. A comprehensive website created to meet the demands of enthusiastic travellers. Our software service, which is based on the Yii version 2 Framework (Yii2 Framework) [1] which is basically consists of Yii Version 2 Basic Framework (Yii2 Basic) [2], Yii Version 2 Advanced Framework (Yii2 Advanced) [3] for the front-end and back-end it is also integrated with ownCloud [4]. It provides a number of features to assist tourists in locating the best tourist attractions and join organized tours. Our software also enable users to share and sync data of their journey like portfolio, pictures and videos with family and friend circle or other adventurers to establishing a lively community of tourism.

The foundation of website application is developed on the Yii2 Basic, that offers a dependable, efficient, affective and interactive environment for web development. We will create essential features like visualization of tour packages and their details, user feedbacks and complains. Yii2 Basic which provides code generating tool which speeds up the overall development process. This guarantees that users can obtain the data they need like information, price plans, to efficiently plan their trips and use the platform with ease.

WWT Application's functionalities are enhanced through the Yii2 Advanced. Yii2 Advanced gives the ability to develop a highly flexible and modular application with abilities like RESTful API building, module-based structure, and intelligent caching techniques. This allows effective integration of various functionality and third-party services, assuring an optimal user experience and enhancing the application's efficiency, security, reliability and scalability to handle growing user base.

Additionally, the ownCloud is integrated with the platform to facilitate users to share and sync data such as photo sharing among trip fellows, friends and family. OwnCloud is an open-source platform that provide on-premises deployment feature. It provides tourists a safe place to upload and exchange their tour videos and photographs. As, WWT application provides ownCloud services, that allowing users to simply upload their data and share them. This feature enriches the platform's community side by encouraging tourists to connect with, engage one another, motivate and build a vibrant network of peoples with similar interest.

WWT Application delivers an exclusively full solution to tourists. This platform is comprehensive and scalable that facilitates booking tours and gives access to major tourism destinations which is empowered by Yii2 Basic and Yii2 Advanced. OwnCloud integration plays a very important role in this application which allows users to share their data.

II. SPECIFICATIONS

This section outlines the specification for the WWT application, aiming to provide a detail overview of user stories by following the requirement elicitation and defining use cases.

User Stories:

1. As a business owner, I want the basic features of the application to be fast and reliable, so that the users can access the application quickly without any disturbance.
2. As a visitor, I want to interact with user-friendly interface where I can see information about popular tourism destinations, tour packages and package details.
3. As a visitor, I need business contact details, so that I can give feedbacks or register complains.
4. As a user, I want to have my personalized account, so that I receive recommendations for tour destinations and tour packages based on my previous tour history and preferences.
5. As a user, I want to be able to book tour packages and avail tour services directly through the application, including transportation, accommodation and to receive real-time

notifications about any changes to my tour plans, such as flight delays or cancellations.

6. As a user, I want to pay for the trip charges through a secure platform such as PayPal, Master Card, or Debit Cards, so that I can process my payments conveniently and securely.
7. As a business owner, I need an open-source file sharing platform with secure and reliable photos and videos sharing features, so that everyone can easily and efficiently share large files with each other.
8. As a user, I want to upload my trip photos to the platform, so that I can capture my tour experiences, and share my trip photos with my friends or fellow travellers.

A. Functional Requirements

- Basic Features (FR-01): Basic features of the application should have a page for visitors to visit Information about popular tourism destinations, tour packages, and package details. Tour Package details should have redirection to a new page with the descriptions, pricing, and availability of tour package information.
- Contact Form (FR-02): Basic features should have a contact form for visitors and complainants to give feedback or register complaints.
- Advanced Features (FR-03): Advanced features should be accessible only to authenticated users. Sign-up page for users to register for authorization to advanced features of the application. Login page for users to get access to advanced features of the application.
- Personalized Accounts (FR-04): Users should be able to manage and customize personalized accounts. The application should provide personalized recommendations for destinations and tour packages using user data and tour history.
- Booking and Notifications (FR-05): Users should be able to book everything directly through the application such as tour packages, transportation and accommodation for tourists. In case of any changes to user booking schedule or updates to their tour plans, the application should send real-time notification to the user.
- Payment Processing (FR-06): Users can process their tour packages payment through a single payment page, which should be integrated with the trusted payment methods. Payment processing should support secure methods like PayPal, MasterCard, and Debit Cards.
- File Sharing (FR-07): Users should be able to upload and download their trip photos on the platform, share them with others, and manage their media content.

B. Non-Functional Requirements

- User Interface: The user interface could be designed with a focus on usability and user experience to cater users of all skill levels. Basic features of the application should be responsive, interactive and highly appealing as well as user-friendly.
- Performance: The application should load the basic features fast and efficiently without any response time, such as popular tourism destinations, tour packages, and their details. Response times for critical operations, such as booking and payment, should be minimal. Large file uploads and downloads should be handled efficiently.
- Security: Critical data including payment information, and transactions should be securely handled and encrypted. Secure payment gateways should be integrated for secure transactions. Access to advanced features should be restricted to authorized users. The file sharing platform should support secure photos and videos sharing among users.
- Reliability: The application shall have high availability to minimize downtime and disruptions. Automated backups shall be performed regularly to ensure data integrity. The file sharing platform should support reliable sharing of photos and videos among users.
- Scalability: The infrastructure should be designed to scale horizontally and vertically to accommodate increasing user demands. The application should be designed to accept upgrades and technology change with minimal efforts.
- Compatibility: The application shall be compatible with popular web browsers and devices, including desktop and mobile platforms.
- Data Privacy: The application shall be in alignment with the data protection laws and make sure user privacy and consent for data processing is in place.

C. Use Cases:

- FR-01-UC-01: Visit Home Page

Precondition: The visitor has access to the internet and the application is running.

Actors: Visitor

Description: How the user access basic features of the application.

Activities: The visitor opens the browser and browses the default URL of the application. The application fetches and displays a list of popular tourism destinations, and tour packages.

- FR-01-UC-02: Tour Package Details

Actors: Visitor

Description: The visitor can explore details of the tour packages in the application.

Activities: The visitor can select a specific tour package to view its details. The application redirects the visitor to a new page dedicated to the selected tour package details. The application displays the description, pricing, and availability details of the tour packages.

- FR-01-UC-03: Search Tour Packages

Actors: Visitor

Description: The visitor can search for specific tour packages based on keywords.

Activities: The visitor enters specific keywords in the search bar. The application filters and displays tour packages that match the entered keywords.

- FR-02-UC-01: Submit Feedback

  Actor: Visitor

  Description: This use case describes the process of a visitor submitting feedback or complaint through the contact form on the website.

  Preconditions: The website is accessible and functional. The contact form is visible and accessible to the visitors.

  Activities: The visitor navigates to the Contact Us page through the navbar in the application. The user selects the option to provide feedback or register a complaint. The system presents the contact form, including fields such as name, email address, title, and description. The user fills in the required information: a) Name: User provides their Full Name. b) Email Address: User enters a valid email address for response purposes. c) Subject: User specifies the subject of their feedback or complaint. d) Description: User types of the details of their feedback or complaint. The user submits the filled-in contact form. The system validates the form data fields and formats. If the data is valid, the system sends a confirmation email to the user's email address with a unique reference number.

- FR-03-UC-01: User Registration

  Actor: User

  Description: This use case describes the process of a visitor signing up for authorization to access advanced features of the application.

  Preconditions: The application is accessible and functional. The user has not registered or logged in before.

  Activities: The user accesses the application's advanced features through the application default domain. The user clicks on the Register link in the navigation bar. The system presents the user with the registration form. The user fills in the required information: a) Username: User chooses a unique username. b) Email Address: User provides a valid email address for communication and account verification. c) Password: User creates a secure password to protect their account.

  The user clicks the register button to submit the form. The system validates the form data: a) Checks for the availability of the chosen username. b) Verifies the email format and uniqueness. c) Ensures the password meets security requirements.

  If the data is valid, the system creates an account for the user and stores their registration details in the database. The system sends a verification email to the provided email address for account activation.

- FR-03-UC-02: User Account Activation

  Actor: Registered User, System Admin

  Description: This use case describes the activation of a user account to login.

  Preconditions: The user has already registered. The user has access to the email address provided. The admin has access to the user management page.

  Activities: The user confirms the verification email sent to the user's provided email address. The system admin approves the user in the database, if the admin ensures that the user is not a spam. The user is activated and able to login to the system.

- FR-03-UC-03: User Login

  Actor: User

  Description: This use case describes the process of an authenticated user logging in to access the advanced features of the application.

  Preconditions: The user has already registered and activated their account. The user has access to the login page.

  Activities: The user navigates to the application's login page. The user enters their registered username or email address and the associated password. The user submits the login credentials. The system validates the entered credentials: a) Checks if the username/email. b) password match an existing account in the database. If the credentials are valid, the system logs the user into the application and grants access to the advanced features.

- FR-04-UC-01: User Profile Management

  Actor: User

  Description: This use case describes the process of a user, managing and customizing their personalized account.

  Preconditions: The user has already registered and logged-in. The user has access to their account settings.

  Activities: The user navigates to the Personalize Account management page. The system displays the user's registered profile information and other relevant details regarding account personalization which will be the default information. The user selects the option to edit their profile. The system presents a form with editable fields to personalize the account according to their interests. The user makes the desired changes to their personalized account data. The user submits the updated profile information. The system validates the modified data: a) Checks for any required fields that may have been left empty. b) Verifies the email format if changes were made to the email address. If the data is valid, the system updates the user's profile information in the database. The system sends a confirmation message to the user that their profile has been successfully updated.

- FR-04-UC -02: Personalized Recommendations

  Actor: User, System

Description: This use case describes the process of the application providing personalized recommendations for destinations and tour packages based on user data and tour history.

Preconditions: The user has already registered and logged in. The user has engaged in previous tour activities, such as booking tours or visiting destinations.

Activities: The user navigates to the Recommendations and Preferences section of the application. The system analyzes the user's tour history, preferences, and any other relevant data. The system uses personalized recommendation engines to generate favored recommendations. The user is presented with a list of recommended destinations or tour packages that align with their interests and past tour behavior. The user can view details of each recommendation, such as destination highlights, and package prices and availability. The user can select a recommended destination or tour package to view more information or proceed with booking and payments.

- FR-05-UC-01: Booking Tour Packages

  Actor: User

  Description: This use case describes the process of an authenticated user booking a tour package directly through the application.

  Preconditions: The user has already registered and logged in. The user has access to the tour packages section of the application.

  Activities: The user navigates to the Tour Packages page of the application's advanced feature. The system presents a list of available tour packages, along with brief descriptions and pricing information. The user selects a specific tour package they are interested in. The system displays detailed information about the selected tour package, including activities, inclusions and availability. The user reviews the package details and decides to proceed with the booking. The user provides the necessary personal information, such as full name, current contact details, start and end date.

- FR-05-UC-02: Real-Time Notifications

  Actor: User

  Description: This use case describes real-time notifications to the user in case of any changes to their booking schedule or updates to their tour plans in the application.

  Preconditions: The user has already registered and logged in. The user has an active booking (e.g., tour package, flights, hotels, or activities).

  Activities: The user completes the booking process for a tour package, flights, hotels, or activities through the application. The system stores the user's booking details in the database. Any changes to the user's booking schedule or updates to their tour plans are monitored by the application. If there are any changes or updates relevant to the user's booking, the system generates real-time notifications. The system sends real-time notifications to the user through their preferred notification channel, such as email, in-app notifications, or SMS.

- FR-06-UC-01: Payment Gateways

  Actor: User

  Description: This use case describes how a user adds the payment details to their account for making payments.

  Preconditions: The user is already registered and logged in. The user has already booked a tour plan.

  Activities: The user navigates to the Payments section of the application. The system presents a form for the user to enter their payment details. The user enter payment details and submits the information for validation. The system verifies the entered data for accuracy and security. If the data is valid, the system adds the payment gateway details to the user's account as a new payment method. The user can now select new payment gateway as their preferred payment method during future transactions.

- FR-06-UC-02: Payment for a Booking

  Actor: User

  Description: This use case describes the process of a user making a payment for a booking using a selected payment method.

  Preconditions: The user is already registered and logged in. The user has completed the booking process and reached the payment section. The user has at least one valid payment method added to their account.

  Activities: The user navigates to the payment section after the booking process. The system displays the total amount to be paid for the booking. The user selects the preferred payment method from their saved payment methods (e.g., credit card, PayPal, MasterCard, or Debit Card). The user confirms the payment details and authorizes the transaction. The payment platform processes the transaction securely. The system receives the payment confirmation and updates the user's booking status to paid in the database. The user receives a payment confirmation message via email.

- FR-07-UC-01: Upload Tour Media Files

  Actor: User

  Description: This use case describes how a user uploads trip media files to the platform to share them with others and manage their media content.

  Preconditions: The user has already registered and logged in to the file sharing platform. The user has upload access in the file management section of their account.

  Activities: The user navigates to the file management section by entering the file sharing default domain name. The system presents an option to upload Media. The user selects the upload option. The system opens a file picker or a drag-and-drop interface,

allowing the user to select multiple media files from their device. The user selects the desired files they wish to upload. The system validates the selected files to ensure they are the correct media files and within the limited file size. If the files are valid, the system begins uploading the media files to the platform's server. The system updates the user's media library with the newly uploaded media files, making them accessible from their account.

- FR-07-UC-02: Download Tour Media Files

  Actor: User

  Description: This use case describes how a user uploads trip media files from the platform to their device.

  Preconditions: The user has already registered and logged in to the file sharing platform. The user has upload access in the file management section of their account.

  Activities: The user navigates to the file management section of their account. The user views the list of photos available in their media library or shared by others. The user selects the photo(s) they want to download. The system initiates the download process, and the user's browser or device prompts to save the photos to a specified location.

- FR-07-UC-03: Sharing Media Files

  Actor: User

  Description: This use case describes the process of a user sharing trip photos with others through the platform.

  Preconditions: The user has already registered and logged in to the file sharing platform. The user has upload access in the file management section of their account. The user has uploaded photos or has access to shared photos on the platform.

  Activities: The user navigates to the file management section of their account. The user views the list of photos available in their media library or shared by others. The user selects the photo(s) they want to share with others. The system provides an option to share the selected photos via email, social media, or by generating a shareable link. The user chooses the sharing method and provides the necessary details (e.g., email addresses, social media handles). The system sends the shared photos to the specified recipients or generates a shareable link that can be accessed by others.

- FR-07-UC-04: Manage Media Content

  Actor: User

  Description: This use case describes how a user manages their media content on the platform, including organizing, editing, and deleting photos.

  Preconditions: The user has already registered and logged in. The user has the proper rights to manage media library.

- Main Flow: The user navigates to the media library. The user views the list of photos available in their media library. The user selects the media they want to manage (e.g., organize, edit, delete). The system provides options to organize the photos into albums or categories. The user selects the desired organization method and assigns photos to albums or categories. The user can edit the photo metadata or add captions to provide context. The user can choose to delete one or more media files from their media library.

III. SPECIFICATIONS

In this section, we will discuss regarding architectural design of our software which includes system architecture, frontend and backend technologies and their interaction, data flow and in the end we will discuss file sharing platform design. We also have illustrated Use Case diagram and Activity diagram which are labeled as Figure 1.0 and Figure 1.1 below.

System Architecture:

System architecture is a concept of the design and construction of systems which includes their components, features, interactions or principles of operation. It takes a very broad view of the system, focusing more on its general organization and less on particular implementation details. The architecture of the system determines how individual parts of the system are combined to achieve certain objectives or satisfy specified functionalities.

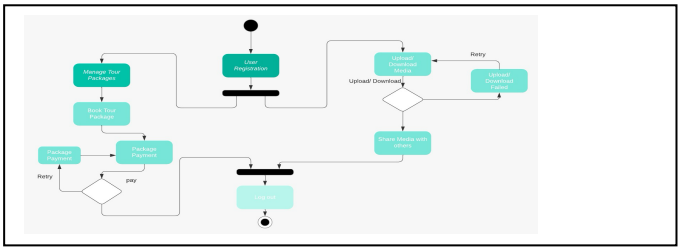

Figure 1.0. Activity Diagram

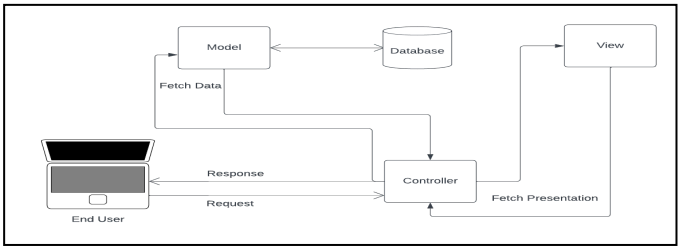

Figure 1.1. MVC Architecture of System Software Diagram

1. Virtual Environment Architecture:

   In WWT Application, we used a VMware to create a virtual machine (VM), we configured that VM by custom configurations like static Internet Protocol (IP) address, domain name, subnet mask and gateway. We also specified hardware requirements and memory allocation like Random Access Memory (RAM), Central Processing Unit (CPU) cores, Hard Disk. To obtain elasticity we used Thin Provisioning with our Hard Disk.

2. Software Architecture:

   In WWT Application, in our VM we need to configure and install operating system, for that we are using open-source, UNIX based operating system, a web server to integrate request-response cycle, object-oriented programming and relational database.

   we needed a well-defined structure which addresses separation of concerns, modularity and reusability. Moreover, we want to excel in the development process by leveraging code quality and collaboration among developers. Keeping in view all these requirements we chose a model-view-controller (MVC) architectural pattern described in the diagram above. MVC works as a fundamental organization for our project. Where data, business logic and rules are dealt by Models, and views show the output on user interface fetched from models. Whereas controllers act as intermediate nodes between users (on views) and models(database). When a user inputs something, controllers manipulate that request and send to models and thereafter views are updated in form of response.

3. Frontend and Backend Technologies

   Our application domain required an interactive and responsive User Interface. To achieve this, we used frontend framework for responsive and mobile-first design, technology for Document Object Model (DOM) manipulation. We will use technologies and libraries designed to improve the experience of developing frontend applications by offering additional features and functionality for basic user interaction, suitable styles and animations if needed. Furthermore, we used well-structured and unique mock-ups to attract tourists to use the platform.

   We needed Rapid Application Development (RAD), MVC Architecture, Rich Feature Set and High Performance so that we can easily convert our audience into potential customers, so we chose object-oriented programming and a framework that enabled us all these features by-default. In addition, we need basic features with efficiency, high availability and responsiveness. In advanced features, we need security, reliability, data integrity and scalability. Considering these features, we further divided the web architectural design into the basic and advanced frameworks.

4. Frontend and Backend Interactions

   In WWT Application, the interaction between frontend and backend is based on separation of concerns, in which frontend handles user interface and presentation logic where backend deals with business logic and data processing. As we are using MVC architectural design for our application which helps to keep separation of concern very clear. We use routing to navigate Uniform Resource Locator (URL) to specific controllers and functions. In the configuration file, we can configure URL patterns for frontend and backend separately. Controllers act as intermediate nodes between client and server side, as they handle user requests like form submission or basic requests and on the basis of URL pattern and route path, it triggers required actions/ functions in the controller. Models serve as data entities and express business logic. Models communicate with backend, like storing or fetching user data from the database and to do data manipulation like create, read, update or delete data. User interfaces (UI) are rendered to the client side by Views, this consist of UIs like contact form or homepage. Their output data is issued by controllers. Except that, we also used Modules, that are used to add modularity in the project. They are reusable components that can be integrated in both the frontend and backend side. They consist of complex user interfaces or functionality. This is also worth considering that we can use APIs or external services for interaction between frontend and backend side.

   Basic features consist of displaying tour packages and their details, a page where user can register their feedbacks and complaints. Whereas, advanced features contain features such as book tour package and package payments. Here we can also integrate real-time data for booking transportation and accommodation options.

5. Data Flow

   We will design Entity Relational Diagram that will list all the entities and their attributes with specific tables and relations. When we talk about data flow between frontend and backend, we use request-response cycle. We need to handle user interaction from frontend of our WWT Application. Users interact with the frontend by performing form submission, clicking buttons or links or requesting Ajax requests. A request goes to the server from the frontend as soon as a user performs an action. Information such as a URL or Hypertext Transfer Protocol (HTTP) protocol will be used in the request. We need a database server to keep our record stored. We will be using a Relational database server because it provides high performance, scalability, community support and keeps its best feature of being cost efficient. The database system should be able to secure critical data such as payment details and user credentials by encrypting it. We should also integrate web-based administration tools for managing databases that database administrators can easily manage and administrate the data. For that we choose a platform that pre-exists. This platform provides best database management for our development environment with interactive user interface where we can see all the data manipulation, data integrity and data encryption.

6. File Sharing Technology

   We need an existing platform where users can share their trip photos, videos and tour-guide. The platform needs to be open-source so we can implement it using a software-as-a-service model. We want on-premises deployment so we will choose a platform that is also compatible with our virtual server tools and resources. It also uses a web server to do http requests and display output to users. We will use a Relational database system to store data. User authentication, file sharing and synchronization should be handled by the platform itself. It should also provide an API to

integrate other features if required. It should also handle access control and user-specific configurations. It should allow administrators to manage user accounts and other permissions. We need an architecture that can do all these features for us, keeping in view that we will design multi-tenant architecture. It is secure and efficient in allowing more than one user to use the service. In this case, user data isolation, control of user quota and enforcement of data privacy and security measures are usually involved. In order to support the high number of users, we must also be able to handle scaling, load balancing and possible deployment of technologies such as containerization or virtualization.

IV. IMPLEMENTATION

In the implementation of our application, we have documented the software and hardware resources that have been or will be implemented in our application environment. We have leveraged cutting-edge technologies and frameworks to develop a robust and scalable solution. Additionally, we have integrated various software tools and libraries to enhance functionality and overcome user's expectation.

1. Hardware Provisioning: Control over the application environment and data privacy are two critical requirements of our client. So, we have chosen to create a virtual machine (VM) in the on-premises datacenter of the business. We accessed our VM through VMware vCenter [1] Server Application (VCSA) with the specific credentials provided to us. We have provisioned a virtual machine with the required resources: a) 16 GB Random Only Memory (ROM) b) 8 GB Random Access Memory (RAM)
2. Software Installation: For successful installation and management of our applications and platforms, we installed Linux, Apache, MySQL, and PHP (LAMP) [5] stack on our VM.
   I. Linux: we have installed Ubuntu 22.04, which is a Linux-based open-source server.
   II. Apache: It is a server used for PHP and MySQL to run.
   III. MySQL: It is a relational database management system used for creating and managing our backend databases.
   IV. PHP: An object-oriented programming language we are using in our application, so we can create, edit, update, and upgrade our application.
   V. Basic Application: We implemented the basic features of our application in Yii basic version 2 framework (Yii2 Basic), for fast and reliable navigation. To install our basic application, we need the following tools:
      - Composer: Dependency management tool for PHP that simplifies the process of installing, managing, and updating libraries or packages used by a project. We used composer to install our Yii2 Basic framework.
   VI. Advanced Application: We implemented advanced features of our application in Yii advanced version 2 framework (Yii2 Advanced), for its security and rich libraries. We are installing Yii Advanced with the composer.

OwnCloud: We installed stable release of ownCloud server version 10.12.1 and updated the latest packages of ownCloud and its dependencies. Apache was configured and required Apache modules were enabled. We created a root user with a password for managing the database of ownCloud. The required privileges were granted to the database to store data of existing user and new users. We created and configured a new database in MySQL for ownCloud to store and manage the user's data and credentials.

Before you begin to format your paper, first write and save the content as a separate text file. Complete all content and organizational editing before formatting. Please note sections A-D below for more information on proofreading, spelling and grammar.

V. VERIFICATION

A. Virtual machine testing

- Installation Testing:

  The virtual machine is successfully installed with the following specifications and configuration:

  The guest additions/tools (if applicable) are installed and function correctly.

- Functionality Testing:

  The virtual machine can access resources on the internet and local network. The VM's storage is accessible, and the users with the right permissions can read and written data without errors. Resource allocation (CPU, memory) performs as intended.

- Performance Testing:

  Expected Result: The VM meets the defined performance benchmarks for CPU, memory, and disk usage. Expected Result: Response times for applications running on the VM are within acceptable ranges.

- Cloning Testing:

  Cloned VM's behave as independent instances and don't interfere with each other.

- Security Testing:

  The VMs isolation mechanisms prevent unauthorized access between different VMs.

- Integration Testing:

  Expected Result: The VM integrates well with management tools, and their functionalities work together as expected. Load balancing mechanisms function correctly in a virtualized environment.

B. Functional Requirements Testing

Test cases in this section are implemented based on the use cases of functional requirements. Test cases are implemented only on critical use cases of the system such as basic features UI (User Interface) efficiency,

authorization, Bookings, and payments. The system is accepted if at least 90 % of the test cases of the critical functionality are successful.

- Test Case (FR-01-UC-01): Visit Home Page

  Start: 2023-07-15 – End: 2023-7-20.

  Test Steps: Open the browser on the test device. Enter the default URL of the application in the browser's address bar. Wait for the application to load and display the home page.

  Expected Results: The browser should open successfully without any errors or warnings.

  The correct default URL of the application should be displayed in the address bar. The application's home page should load without any errors or delays. The home page should display a list of popular tourist destinations and Tour packages that are available for booking or exploration.

  Test Environment:

  (1) *Device: Windows PC.*
  (2) *Browser: Chrome v92.*
  (3) *Internet Connection: Wi-Fi with VPN connection to the Client server.*
  (4) *Application URL: 192.168.67.42/basic*

  Test Results: The browser opens successfully, and no errors or warnings are encountered during the process. The correct default URL "192.168.67.42/basic" is displayed in the browser's address bar. The application's home page loads within an acceptable time frame, and no errors are observed during the loading process. The home page displays a list of popular tourism destinations, along with various attractive tour packages available for booking or exploration.

  Overall Test Result: The test case 'FR-01-UC-01' has passed successfully.

- Test Case (FR-03-UC-01): User Registration for Advanced Features

  Start: 2023-07-30 – End: 2023-08-10.

  Test Steps: Open the browser on the test device. Enter the application URL in the browser's address bar. Press Enter or click on the "Go" button to access the application. Locate and click on the "Register" link on the application's homepage. Fill in the required information on the registration form: a) Choose a unique username for identification. b) Provide a valid email address for communication and account verification. c) Create a secure password to protect the account. Click on the "Register" button to submit the filled-in registration form. Wait for the system to validate the form data and process the registration request. Check if the system performs the following validations correctly: a. Verify the availability of the chosen username. b. Validate the email format and uniqueness. c. Ensure the password meets the security requirements. Verify that if the form data is valid, the system creates an account for the user and stores their registration details in the database. Check the user's email inbox for a verification email from the application for account activation.

  Expected Results: The browser should open successfully without any errors or warnings. The application's homepage should load without any errors or delays. The "Register" link should be easily visible and accessible on the homepage. User registration form should be presented to the user with fields for username, email address, and password. The user should be able to fill in the required information without any issues or restrictions. The register button of the user registration form should work without any errors or delays. The system should perform the following validations accurately: a. Confirm that the chosen username is available and not already taken by another user. b. Validate that the email address is in the correct format and not already associated with an existing account. c. Ensure that the password meets the application's security requirements (e.g., minimum length, uppercase, lowercase, and special characters).If the form data is valid, the system should successfully create an account for the user and store their registration details in the database. The user should receive a verification email in their provided email address for account activation.

  Test Environment:

  i) Device: Windows PC.
  ii) Browser: Chrome v92.
  iii) Internet Connection: Wi-Fi with VPN connection to the Client server.
  iv) Application URL: 192.168.67.42/basic

  Test Data:

  - Username: unique_username
  - Email Address: test@example.com
  - Password: Test@123

  Test Results:

  The browser opens successfully, and no errors or warnings are encountered during the process. The correct application URL '192.168.67.42/basic' is displayed in the browser's address bar. The application's homepage loads within an acceptable time frame, and no errors are observed during the loading process. The "Register" link is clearly visible and clickable on the homepage. The registration form is presented to the user with appropriate fields for username, email address, and password. The user successfully fills in the required information: unique_username, test@example.com, and Test@123.The user clicks on the "Register" button without any errors or delays. The system performs accurate validations for the provided data: a. The chosen username "unique_username123" is available and not taken. b. The email address "test

  Overall Test Result: The test case 'FR-01-UC-01' has passed successfully.

- Test Case: FR-05-UC-01: Booking Tour Packages

  Start: 2023-08-10 – End: 2023-08-20.

  Preconditions: The user has already registered and logged in. The user has access to the tour packages section of the application.

  Test Steps: Log in to the application using valid credentials (username and password). Locate and navigate to the "Tour Packages" section of the application's advanced features. Verify that the Tour Packages page loads without any errors or delays and displays a list of available tour packages with brief descriptions and pricing information. Select a specific tour package from the list that the user is interested in by clicking on it. The system should display detailed information about the selected tour package, including activities, inclusions, availability, and exclusion details. Review the package details provided by the system and ensure that they match the selected tour package.

  Decide to proceed with the booking of the selected tour package. Provide the necessary personal information in the booking form, including

  i) Full name

  ii) Current contact details (phone number or email address).

  iii) Start date of the tour.

  iv) End date of the tour.

  v) Any extra information or special requirements (if applicable) Submit the booking form with the provided information.

  Expected Results: The login process should be successful, and the user should be redirected to their account dashboard or homepage. The "Tour Packages" section should be easily accessible, and the page should load without any errors or delays. The Tour Packages page should display a list of available tour packages with accurate descriptions and pricing information. Clicking on a specific tour package should lead to a detailed page with accurate information about the selected package. The detailed information displayed for the selected tour package should be complete and match the actual details of the package. The user should be able to review the package details without any issues or restrictions. The user should decide to proceed with the booking without encountering any errors or difficulties. The booking form should allow the user to provide their personal information, including full name, contact details, start date, end date, and any extra information, without any issues. Clicking on the "Submit" button should not result in any errors, and the booking form data should be successfully processed.

  Test Environment:

  i) Device: Windows PC.

  ii) Browser: Chrome v92.

  iii) Internet Connection: Wi-Fi with VPN connection to the Client server.

  iv) Application URL: 192.168.67.42/Advanced

  Test Data:

  Tour Package Selected: "Exotic Island Getaway"

  Full Name: Aziz Rahman

  Contact Details: aziz.atal@example.com, +49 (123) 456-7890-4

  Extra Information: None

  Test Results: This feature has not implemented yet.

- Test Case: FR-07-UC-01: Download Trip Media Files

  Start: 2023-08-20 – End: 2023-08-25.

  Preconditions: The user has already registered and logged in to the file sharing platform.

  The user has upload access in the file management section of their account.

  Test Steps: Log in to the file sharing platform using valid credentials (username and password). Locate and navigate to the "File Management" section of the user's account. Verify that the File Management page loads without any errors or delays and displays a list of media files, including photos, available in the user's library or shared by others. Select one or multiple photos from the list that the user wants to download by checking the appropriate checkboxes next to each photo. Click on the "Download" button or a similar download icon to initiate the download process for the selected photos. The user's browser or device should prompt to save the photos to a specified location on the user's device. Choose a location to save the downloaded photos (e.g., a specific folder on the desktop) and click on the "Save" button or similar action to complete the download.

  Expected Results:

  The login process should be successful, and the user should be redirected to their account dashboard or homepage. The "File Management" section should be easily accessible, and the File Management page should load without any errors or delays. The File Management page should display a list of media files, including photos, available in the user's library or shared by others, without any issues. The user should be able to select one or multiple photos from the list by checking the appropriate checkboxes without encountering any errors or restrictions. Clicking on the "Download" button or download icon should initiate the download process for the selected photos without any errors. The user's browser or device should prompt to save the downloaded photos to a specified location on the user's device, such as the desktop or a designated folder. The user should be able to choose a location to save the downloaded photos and successfully complete the download process without any issues.

  Test Environment:

- Device: Windows PC
- Browser: Chrome v92
- Internet Connection: Wi-Fi
- File Sharing Platform URL: https://www.examplefilesharing.com
- Test Data:
  (a) Selected Photos: "beach.jpg, "mountain.jpg," "cityscape.jpg"
  (b) Download Location: Desktop folder named "TripPhotos"

Test Results:

The login process is successful, and the user is redirected to their account dashboard. The "File Management" section is easily accessible, and the File Management page loads without any errors or delays. The File Management page displays a list of media files, including photos ("beach.jpg," "mountain.jpg," "cityscape.jpg"), available in the user's library or shared by others, without any issues. The user successfully selects multiple photos ("beach.jpg," "mountain.jpg," "cityscape.jpg") from the list by checking the appropriate checkboxes without encountering any errors or restrictions. Clicking on the "Download" button or the download icon successfully initiates the download process for the selected photos ("beach.jpg," "mountain.jpg," "cityscape.jpg") without any errors. The user's browser or device prompts to save the downloaded photos ("beach.jpg," "mountain.jpg," "cityscape.jpg") to a specified location (e.g., the desktop) on the user's device without any issues. The user successfully chooses the "Trip Photos" folder on the desktop as the download location and clicks on the "Save" button to complete the download process without any issues.

C. Code testing

a) Code Review:

Manual inspection of the source code by peer developers. During code reviews, the reviewer carefully examines the code to identify issues such as syntax errors, logical flaws, and coding practices. This process helps to catch problems early in the development process.

b) Unit Testing:

Testing individual units (functions, methods, and classes) of code in isolation to ensure they produce the expected output for a given set of inputs. Automated unit tests are implemented to detect regressions and provide immediate feedback to developers when code changes introduce defects.

D. Compatibility testing

a) Browser Compatibility Testing: Test the web application on popular web browsers such as Google Chrome, Mozilla Firefox, Apple Safari, Microsoft Edge, and Internet Explorer (if still supported). Verify that the application's functionality, layout, and design remain consistent and functional across different browsers.

b) Operating System Compatibility Testing: Test the web application on various operating systems, including Windows, macOS, Linux, iOS, and Android. Ensure that the application works seamlessly on different operating systems without any critical issues.

c) Device Compatibility Testing: Test the web application on different devices like desktops, laptops, tablets, and smartphones. Verify that the application adapts well to different screen sizes and resolutions

E. UI testing

a) Functional Testing:

Verify that all UI elements (buttons, links, forms, etc.) are functioning as intended and lead to the correct pages or perform the expected actions. Test form validations to ensure that error messages appear appropriately when users enter invalid data. Confirm that user inputs are properly processed and produce the desired outcomes.

b) Layout and Design Consistency:

Verify that the application's layout, color schemes, fonts, and design elements remain consistent across different pages and screen resolutions. Check for alignment, spacing, and formatting issues to ensure a polished and professional appearance.

c) Responsive Design Testing:

Test the application on various devices with different screen sizes (desktops, laptops, tablets, smartphones) to ensure responsiveness. Verify that the application adjusts and renders appropriately to fit the device's screen.

d) Cross-Browser Testing:

Test the application on multiple web browsers (e.g., Google Chrome, Mozilla Firefox, Apple Safari, Microsoft Edge, Internet Explorer) to ensure compatibility and consistent appearance.

e) Navigational Testing:

Test the application's navigation to ensure that users can easily move between different pages and sections. Verify that all links and buttons lead to the correct destinations.

f) Error Handling and Messages:

Verify that appropriate error messages are displayed when users encounter errors or validation issues. Ensure that error messages are clear and helpful in guiding users to resolve the problems.

F. Application Security testing

1. Vulnerability Assessment:

Conduct a vulnerability assessment to identify potential security flaws, such as SQL injection, Cross-Site Scripting (XSS), Cross-Site Request Forgery (CSRF), etc. Utilize automated scanning tools to scan the application for known vulnerabilities.

2. Penetration Testing (Pen Testing):

Perform penetration testing to simulate real-world attacks and attempt to exploit identified vulnerabilities. Ethical hackers attempt to gain unauthorized access to the application and assess its resilience to attacks.

3. Authentication and Authorization Testing:

   Verify that the authentication mechanism is secure, enforcing strong passwords and preventing common authentication bypass techniques. Test authorization to ensure users can only access appropriate resources and actions based on their roles and permissions.

4. Input Validation Testing:

   Validate that user inputs are sanitized correctly to prevent injection attacks (e.g., SQL injection, LDAP injection). Ensure that user-provided data is validated and properly encoded to prevent XSS attacks.

5. Session Management Testing:

   Test session management mechanisms to ensure that session tokens are adequately protected, and session timeouts are enforced. Verify that session IDs are not exposed in URLs or easily guessable.

6. Secure Communication (SSL/TLS):

   Check that the application uses SSL/TLS to encrypt data transmission between the client and the server. Verify that certificates are valid, not expired, and correctly configured.

7. Error Handling and Information Leakage:

   Evaluate how the application handles errors and ensures that detailed error messages are not leaked to users or attackers. Prevent disclosing sensitive information that could aid attackers in their efforts.

8. Security Headers and Configuration:

   Ensure that security-related HTTP headers (e.g., Content Security Policy, X-XSS-Protection, X-Content-Type-Options) are correctly configured. Verify that security features, such as HTTP Strict Transport Security (HSTS), are in place where applicable.

9. File Upload and Download Testing:

   Test file upload functionalities to ensure that users can only upload allowed file types and that uploaded files are thoroughly scanned for malware. Verify that file download mechanisms do not expose sensitive files or directories.

VI. VARIFICATION

A. User Acceptance Testing (UAT)

1. FR-01: Basic Features:

   Test Scenarios: To test that the home page of the basic features displays the information (popular tourism destinations and tour packages) is accurate and in valid format. Test that clicking on a specific tour package redirects the user to a new page with detailed descriptions, pricing, and availability. Ensure the tour package information is correct and consistent with the actual offerings.

   Execute Test Cases: Successfully executed by client.

   Verify Expected Results: All functionalities working as expected.

   Report Defects: No defects

   Fixed Issues: No fixed Issues.

2. FR-02: Contact Form:

   Test Scenarios: Test the contact form as a visitor, by submitting valid feedback and complaints information. Verify that the form is submitted successfully, and the user receives a confirmation message via email. Test contact form with invalid inputs to ensure proper error handling. Check that user complaints and feedback are forwarded properly for further processing.

   Execute Test Cases: Successfully executed by client.

   Verify Expected Results: All functionalities working as expected.

   Report Defects: No defects

   Fixed Issues: No fixed Issues.

3. FR-03: Advanced Features:

   Test Scenarios: Test the sign-up page to ensure that the users can register for authorization to advanced features. Verify that only authenticated users can access advanced features. Test the login page to ensure users can access advanced features after authentication. Ensure that unauthorized users are restricted from accessing all advanced features. Execute Test Cases: Successfully executed by client. Verify Expected Results: All functionalities working as expected.

   Report Defects: No defects

   Fixed Issues: No fixed Issues.

4. FR-04: Personalized Accounts:

   Test Scenarios: Test the account management functions, such as updating personal information and preferences are working properly. Verify that personalized recommendations for destinations and tour packages are provided based on user data and tour history. Check that the recommendations are relevant and aligned with the user's interests and preferences.

   Execute Test Cases: Not implemented.

   Verify Expected Results: Not implemented.

   Report Defects: No defects

   Fixed Issues: No fixed Issues.

5. FR-05: Booking and Notifications:

   Test Scenarios: Test the booking process for tour packages, flights, hotels, and activities. Verify that bookings are confirmed, and the user receives booking details via email or notification. Test the real-time notification system for updates to user booking schedules or changes to tour plans. Confirm that users receive notifications promptly and accurately.

Execute Test Cases: Not implemented.

Verify Expected Results: Not implemented.

Report Defects: No defects

Fixed Issues: No fixed Issues.

6. FR-06: Payment Processing:

Test Scenarios: Test the payment processing system using trusted platforms like PayPal, MasterCard, and Debit Cards. Make sample payments to verify that transactions are successful and secure. Check for any errors or discrepancies during payment processing.

Execute Test Cases: Not implemented.

Verify Expected Results: Not implemented.

Report Defects: No defects

Fixed Issues: No fixed Issues.

7. FR-07: File Sharing:

Test Scenarios: Test the file upload feature to ensure users can upload trip photos and other media content. Verify that uploaded files are correctly stored and associated with the user's account. Test the download and sharing functionality to ensure files can be accessed and shared with others. Check that users can manage and organize their media content effectively.

Execute Test Cases: Successfully executed by client.

Verify Expected Results: Users cannot login to the file sharing platform.

Report Defects: User access is restricted.

Fixed Issues: No fixed Issues.

B. Design and Layout

Validate that the website design aligns with the client's branding guidelines and visual preferences. Check that the layout is user-friendly and intuitive, with proper placement of elements.

C. Content Verification

Ensure that all website content, including text, images, videos, and other media, accurately represents the client's business and adheres to their tone and style.

D. Responsiveness and Compatibility

Test the website on various devices and screen sizes to ensure it is responsive and displays correctly on desktops, laptops, tablets, and smartphones. Verify compatibility with different web browsers.

E. Link and Navigation Check

Validate all internal and external links to ensure they lead to the correct pages and resources. Check the navigation flow to ensure users can access information easily.

F. Form Validation

Test all forms on the website to ensure they have proper validation and error handling. Verify that form submissions are processed correctly.

G. Performance Testing

Test the website's loading speed and performance to ensure it meets the client's performance expectations.

VII. REFERENCES

[1] "Get Started," yiiframework, 2008. [Online]. Available: https://www.yiiframework.com/. [Accessed 25 June 2023].

[2] "The Definitive Guide to Yii 2.0," yiiframework, 2008. [Online]. Available: https://www.yiiframework.com/doc/guide/2.0/en/start-installation.

[3] "Yii 2 Advanced Project Template," yiiframework, [Online]. Available: https://www.yiiframework.com/extension/yiisoft/yii2-app-advanced/doc/guide/2.0/en/start-installation.

[4] "ownCloud GmbH," ownCloud, 3 April 2020. [Online]. Available: https://owncloud.com/. [Accessed 7 July 2023].

[5] J. Camisso, "How to install PHP 8.1 And set up a local development environment on Ubuntu 22.04," DigitalOcean, 4 May 2022. [Online]. Available: https://www.digitalocean.com/community/tutorials/how-to-install-php-8-1-and-set-up-a-local-development-environment-on-ubuntu-22-04. [Accessed 3 July 2023].

[6] R. Raghuram, "VMware vSphere Documentation," vmware, [Online]. Available: https://docs.vmware.com/en/VMware-vSphere/index.html. [Accessed 4 July 2023].