| Fontys University of Applied Sciences

![A bus on the road

Description automatically generated]()

**Himalayan Bus****: Project Plan**

Bus tickets reservation system

Himal Aryal [2954222]

2023

Table of Contents

[1. Project assignment 3](#_Toc146550851)

[1.1 Context 3](#_Toc146550852)

[1.2 Problem Statement: 3](#_Toc146550853)

[1.3 Goal of the project 3](#_Toc146550854)

[1.4 Scope and preconditions 3](#_Toc146550855)

[1.4.1 Inside Scope: 3](#_Toc146550856)

[1.4.2 Outside Scope: 4](#_Toc146550857)

[1.5 Strategy 4](#_Toc146550858)

[1.6 Research questions and methodology 4](#_Toc146550859)

[1.6.1 Research Question 1: 4](#_Toc146550860)

[Methodology (Dot Framework): 5](#_Toc146550861)

[1.6.2 Research Question 2: 5](#_Toc146550862)

[Methodology (Dot Framework): 5](#_Toc146550863)

[1.6.3 Research Question 3: 5](#_Toc146550864)

[1.7 End products 6](#_Toc146550865)

[2. Activities and time plan 7](#_Toc146550866)

[2.1 Phases of the project 7](#_Toc146550867)

[2.1.1 Project Initiation Phase: 7](#_Toc146550868)

[2.1.2 System Design and Planning Phase: 7](#_Toc146550869)

[2.1.3 Development and Implementation Phase: 7](#_Toc146550870)

[2.1.4 Testing and Quality Assurance Phase: 8](#_Toc146550871)

[2.1.5 Deployment and User Training Phase: 8](#_Toc146550872)

[2.1.6 Operations and Monitoring Phase: 8](#_Toc146550873)

[2.1.7 Data Analysis and Decision-Making Phase: 8](#_Toc146550874)

[2.1.8 Project Handover Phase: 8](#_Toc146550875)

[2.1.9 Evaluation, Reflection, and Closure Phase: 8](#_Toc146550876)

[2.2 Time plan and milestones 8](#_Toc146550877)

[2.2.1 Sprint 1 8](#_Toc146550878)

[2.2.2 Sprint 2 8](#_Toc146550879)

[3. Testing strategy and configuration management 10](#_Toc146550880)

[3.1 Testing strategy 10](#_Toc146550881)

[3.2 Test environment and required resources. 11](#_Toc146550882)

[3.3 Configuration management 11](#_Toc146550883)

[4. Risk 13](#_Toc146550884)

[4.1 Risk and mitigation 13](#_Toc146550885)

**Version history**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Version** | **Date** | **Author(s)** | **Changes** | **State** |
| 1 | 2023/09/22 | Himal | Project plan started | completed |
| 2 | 2023/10/12 | Himal | Research part has been removed,  deliverables are duplicate=ed therefore its also removed,  project agile has been divided into 2 phases,  Planning needs to be consider after meeting | Completed almost/ |
|  |  |  |  |  |

# Project assignment

## Context

"Himalayan Bus" is a bus service provider operating in a European region. They are committed to improving their services and customer experience by embracing technology.

## Problem Statement:

The current absence of an efficient online booking platform presents several challenges, including:

1. **Inefficient Booking Process:** The absence of an online booking system forces customers to rely on traditional methods, leading to time-consuming and cumbersome ticket reservations, often resulting in frustration and dissatisfaction.
2. **Operational Inefficiencies:** Manual ticketing processes are labor-intensive and prone to errors, leading to operational inefficiencies for Himalayan Bus. This includes ticketing errors, overbooking, and difficulties in managing passenger data.
3. **Limited Revenue Generation:** The lack of an online presence limits the bus service provider's ability to tap into the potential of online sales and marketing. This restricts revenue generation opportunities and hampers the company's competitiveness in the market.
4. **Customer Dissatisfaction:** Modern travelers expect the convenience of booking tickets online. The absence of this feature may lead to customer dissatisfaction and loss of potential customers to competitors offering online booking options.
5. **Data Utilization Challenges:** The company is unable to harness the power of data-driven decision-making due to the absence of an integrated reservation system. This hinders their ability to analyze customer preferences, optimize routes, and make informed operational decisions.

## Goal of the project

The goal of the bus tickets reservation system (“Himalayan Bus”) project is to create an efficient online platform for booking and purchasing bus tickets. This project aims to provide customers with a user-friendly experience, optimize the bus service provider's operations, and generate revenue by leveraging Information and Communication Technology (ICT). The system adds value by enhancing customer satisfaction, improving efficiency, and fostering data-driven decision-making, all while positioning the company as a competitive and customer-centric service provider.

## Scope and preconditions

### Inside Scope:

1. User Profile Management: Within the project's scope, user profile management includes user registration, login, and basic user information maintenance. Users can update their personal details, view their booking history, and reset their passwords.
2. Bus Route and Schedule Management: The project includes features that allow administrators to manage bus details, such as Bus Name, Bus Route, Bus Type (AC/non-AC), seats, departure time, and arrival time. Admins can also manage the available routes.
3. Feedback and Reviews: Gathering and displaying user feedback and reviews for bus journeys is part of the project's scope. Users can provide feedback on their travel experiences, and this information will be accessible to other users for reference.

### Outside Scope:

1. Payment Gateway Integration: The integration of specific payment gateways for processing ticket payments is typically outside the initial project scope. This integration involves financial transactions and requires careful consideration of security and compliance.
2. Real-Time Bus Tracking: Implementing real-time tracking of buses and providing live updates on bus locations and delays is typically a complex and resource-intensive feature that is often considered for future enhancements rather than the initial project.
3. Advanced Analytics and Reporting: In-depth analytics and reporting functionalities, such as complex data analysis, revenue forecasting, or route optimization based on historical data, may fall outside the initial project scope. These advanced features often require more extensive development and data processing.

## Strategy

The Agile approach is a strong fit for the bus ticket reservation project for several reasons:

1. Changing Requirements: In a project like this, requirements can evolve as customer needs and market conditions change. Agile allows for flexibility in accommodating these changes without disrupting the entire project plan.
2. User-Centric Focus: The project's success depends on providing a user-friendly and efficient platform. Agile's emphasis on customer feedback ensures that user preferences are incorporated into the design and functionality of the system.
3. Incremental Development: Agile enables the delivery of working features in short cycles. This means that even in the early stages of the project, functional components can be delivered to users, providing value and allowing for early validation.
4. Collaboration: The bus ticket reservation system involves various stakeholders, including developers, administrators, and end-users. Agile promotes collaboration and regular communication among these groups to ensure everyone's needs are addressed.
5. Continuous Improvement: Agile's iterative approach allows the team to learn from each sprint and make improvements, enhancing the quality of the system over time.

In summary, the Agile approach is a well-suited strategy for the bus ticket reservation project because it aligns with the project's need for adaptability, user-centric design, incremental development, collaboration, and continuous improvement. It provides a framework that can accommodate changing requirements and deliver a more successful and customer-focused product.

## End products

A screenshot of a computer

Description automatically generated

# Activities and time plan

## Phases of the project

In the case of the "Himalayan Bus" bus ticket reservation system project, here are the main phases:

Initiation Phase:

This phase would still exist and involve setting up the project, creating a product backlog, and defining the project's vision and goals.

Sprints/Iterations:

In Agile, the project work is divided into a series of sprints or iterations, each typically lasting 2-4 weeks. During each sprint, a cross-functional works on a set of prioritized user stories or features from the product backlog. The work is divided into short, time-boxed cycles, and at the end of each sprint, a potentially shippable product increment is delivered.

## Time plan and milestones

The artifacts listed in each sprint represent the work items to be completed during that sprint. Let's review each sprint's artifacts:

### Sprint 1

1. Project plan.
2. Initial product backlog.
3. Backend: first setup of RESTful API.
4. Layering, dependency inversion and injection using Spring.
5. CI/CD environment initialization.

### Sprint 2

1. Architecture constraints and design decisions.
2. C4 Model diagrams with context/explanations for level 1 (system context), 2 (container) and 3 (component).
3. Backend: CORS configuration in-place for controllers.
4. Initial Frontend setup.

|  |  |  |  |
| --- | --- | --- | --- |
| **Phasing** | **Effort** | **Start date** | **Finish date** |
| Sprint 1 | High | 9/15/2023 | 9/22/2023 |
| Sprint 2 | High | 09/23/2023 | 10/12/2023 |
| Sprint 3 |  |  |  |
| Sprint 4 |  |  |  |
| Sprint 5 |  |  |  |
| Sprint 6 |  |  |  |

# Testing strategy and configuration management

## Testing strategy

A diagram of testing

Description automatically generatedIn the testing strategy for the bus ticket reservation system project, I will employ a multi-level approach to ensure the reliability and quality of the software. The following testing levels will take place:

**Unit Testing:**

* **Scope:** Unit testing will be performed on individual components and functions of the system, such as specific modules, classes, or methods.
* **Automation:** Unit tests will be automated using testing frameworks like JUnit or NUnit.
* **Goals:** Achieve high code coverage (e.g., 80% or higher) for unit tests to ensure that critical components are thoroughly tested.

**Integration Testing:**

* **Scope:** Integration testing validates the interactions between different parts of the system, including the frontend and backend.
* **Automation:** Some integration tests can be automated, especially those involving APIs or data integration.
* **Goals:** Identify and address any issues related to data exchange, communication, and overall system integration.

**System Testing:**

* **Scope:** System testing evaluates the entire bus ticket reservation system as a whole. It focuses on end-to-end scenarios and user journeys.
* **Automation:** While many system tests will be manual to simulate real user interactions, some regression tests can be automated.
* **Goals:** Ensure that the system meets all functional and non-functional requirements, including usability, performance, and security.

**Acceptance Testing:**

* **Scope:** AT involves real users (or simulated users) testing the system to determine if it meets their needs and expectations.
* **Automation:** Typically, UAT is performed manually by users to mimic their experience realistically.
* **Goals:** Obtain user feedback and ensure that the system aligns with user requirements.

**Justification:**

* **Comprehensive Coverage:** This multi-level testing strategy ensures comprehensive coverage, from unit-level validation to end-to-end system testing.
* **Early Issue Detection:** Unit and component testing help detect and address issues at an early stage of development, reducing the cost of fixing defects later.
* **User-Centric Approach:** User acceptance testing ensures that the system meets user expectations and is aligned with real-world usage.
* **Non-functional Validation:** Non-functional testing guarantees that the system not only functions correctly but also perform well, is secure, and can scale as needed.
* **Automation Benefits:** Automation helps in regression testing and ensures that code changes do not introduce new defects.
* **Flexibility:** Manual testing allows for exploratory testing and the identification of unexpected issues.

## Test environment and required resources.

**Test Environment:**

A diagram of a testing process

Description automatically generatedFor the bus ticket reservation system project, I envision a test environment that follows a typical DTAP (Development, Testing, Acceptance, Production) pipeline. This approach ensures that each phase of the project is thoroughly tested before moving to the next stage.

**CI/CD Pipeline:**

To streamline development and testing, a Continuous Integration/Continuous Deployment (CI/CD) pipeline will be established. The CI/CD pipeline automates the build, testing, and deployment processes. It helps in ensuring that code changes are integrated and tested frequently, leading to faster development cycles and more reliable releases.

## Configuration management

In the bus ticket reservation system project, a robust configuration management approach will be implemented using Git as the version control system. Here are the key aspects of the version management strategy:

**Tooling:** Git will be used as the primary version control system, and a central Git repository will be set up to manage the project's source code and related files. Git provides powerful branching, merging, and tracking capabilities.

**Branching Strategy (Single Branch - master):**  
In a simplified individual project, using only one branch (typically the **master** branch) is a straightforward approach.

**Master Branch:** The **master** branch serves as the primary branch for my project. All development work, including new features, bug fixes, and improvements, is done directly in this branch.

**Promotion, Release, and Baseline Strategy:**

* **Promotion:** In this simplified approach, code moves progressively forward within the master branch, from development to stable and production-ready states.
* **Release Strategy:** For significant milestones or versions, I can create version tags within the main branch to mark specific releases. These tags act as snapshots of my project's state at those points in time.
* **Baseline Strategy:** While not formally designated as baselines, my version tags effectively serve as baselines for tracking the project's progression.

# Risk

## Risk and mitigation

|  |  |  |  |
| --- | --- | --- | --- |
| **Risk** | **Impact** | **Prevention activities** | **Mitigation activities** |
| 1. **Technical Challenges** | High | * Conduct thorough research and prototyping before starting development. * Seek advice from online communities and experts when facing technical difficulties. | * Allocate extra time in the project schedule to account for unforeseen technical challenges. * Maintain a list of alternative solutions or approaches in case of technical roadblocks. |
| 1. **Scope Creep** | Moderate | * Clearly define and document the project's scope and objectives. * Establish a change control process for scope changes. | * Review and assess all proposed scope changes before implementation. * Communicate with stakeholders to manage expectations when scope changes are approved. |
| 1. **Quality Issues** | High | * Define clear quality standards and criteria for deliverables. * Conduct regular quality checks and inspections throughout the project. | * Address any quality issues promptly and implement corrective actions. * Perform root cause analysis to prevent similar issues in the future. |