**Project Plan**

**Ticketpass**

|  |
| --- |
| **Date : 6th September 2023** |
| **Version : 1.0** |
| **State : Early Development** |
| **Author : Hristo Ganchev** |

#### Version history

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Version** | **Date** | **Author(s)** | **Changes** | **State** |
| 1.0 | 6th September 2023 | Hristo Ganchev | Initiation of the project. | Early development |
|  |  |  |  |  |
|  |  |  |  |  |

**Distribution**

|  |  |  |
| --- | --- | --- |
| **Version** | **Date** | **Receivers** |
|  |  |  |
|  |  |  |

**Contents**

[1. Project assignment 4](#_Toc145489028)

[1.1 Context 4](#_Toc145489029)

[1.2 Problem 4](#_Toc145489030)

[1.3 Goal of the project 4](#_Toc145489031)

[1.4 Scope and preconditions 4](#_Toc145489032)

[1.5 Strategy 5](#_Toc145489033)

[1.6 Research questions and methodology 5](#_Toc145489034)

[1.7 End products 5](#_Toc145489035)

[2. Project organisation 6](#_Toc145489036)

[2.1 Stakeholders and team members 6](#_Toc145489037)

[2.2 Communication 6](#_Toc145489038)

[3. Activities and time plan 7](#_Toc145489039)

[3.1 Phases of the project 7](#_Toc145489040)

[3.2 Time plan and milestones 7](#_Toc145489041)

[4. Testing strategy and configuration management 8](#_Toc145489042)

[4.1 Testing strategy 8](#_Toc145489043)

[4.2 Test environment and required resources 8](#_Toc145489044)

[4.3 Configuration management 8](#_Toc145489045)

[5. Risk and mitigation 9](#_Toc145489046)

[5.1 Risk and mitigation 9](#_Toc145489047)

# Project assignment

## Context

The focus of the project is to create a website for purchasing concert tickets. There will be an administrator side and a user (buyer) side.

## Problem

The main problem is that usually, the user would need to buy tickets from separate venue websites, having all of their tickets in different places and making it more difficult for themselves. They will need to create a lot of accounts, having the chance they will lose access to one of them or forget which ticket is on which account.

## Goal of the project

Ticketpass is a website for buying concert tickets. You make an account in the website, select a concert you would like to attend, buy a ticket, and you are ready to go! The ticket is stored in your account and you can buy as many tickets for as many concerts as you want. All of the tickets that you have purchased can be accessed in one place. There is also an administrator who has access to all ticket sales and information and they can also add events.

The idea of this project is to create a website where the user has an easy access to many concerts that they can buy tickets for. The user can see all of their tickets in one place.

## Scope and preconditions

|  |  |
| --- | --- |
| **Inside scope:** | **Outside scope:** |
| 1. Ability to obtain concert tickets | 1. Ability to buy artist merchandise |
| 1. All of the tickets will be stored in one place | 1. Ability to buy venue memberships |
| 1. Administrators will be able to see all of the ticket purchases made from all users and manage concerts (add/delete/update) | 1. Ability for artists to add their concerts to the website |
| 1. User will also receive the tickets by email | 4 Ability to pay for the tickets |
| 1. Update personal information | 5 Ability to obtain VIP tickets |
| 1. Ability for the admin to delay or cancel a concert | 6 User cannot refund or resell their ticket |

## Strategy

The agile methodology will be used in the project implementation. Since the project is divided into six sprints, feedback will be asked for and received constantly, and the feedback will then be applied into the project in order to make it as perfect as possible and satisfy the stakeholders.

## Research questions and methodology

What do the stakeholders require? How to connect the frontend to the backend? What’s SonarQube and how to work with it? How to make the login service JWT based? How to work with web tokens? How to implement Minimum viable product features? What are OWASP top 10 security risks?

To reach to the final goal of the project, the developer will constantly keep in touch with the stakeholders in order to make sure that all of the features required have been implemented.

## End products



# Project organisation

## Stakeholders and team members

The stakeholders of the project are the teachers and the project team consists of one person: Hristo Ganchev. The stakeholders are the ones who put the requirements and act as clients to the project.

## Communication

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Abbreviation** | **Role and functions** | **Availability** |
| Hristo Ganchev | H.D. Ganchev | The full-stack developer of the project. He is responsible for the design, the frontend, backend, testing and documentation. | The developer is available during the weekdays between office hours. He can be contacted via phone, email, Microsoft Teams, etc. |

The communication with stakeholders can be made either face-to-face or via Microsoft Teams. It should be established at least once a week in order to receive feedback, reflect on it, apply it in the software solution and repeat until finalization.

# Activities and time plan

## Phases of the project

The first phase of the project is the planning. First, the idea should be discussed with the teachers, and whether it is satisfactory enough to initiate development. Next is the beginning of the project. Every idea is turned into a bullet point and then thought on whether to be added to the project or not. Every three weeks, evaluation is made, whether the project development is on track or not. After five evaluations the finished version of the project must be presented.

## Time plan and milestones

The project is divided into six sprints of three weeks. In the end of each sprint, the current version of the project must be presented and evaluated by the stakeholders (the teachers).

The provided start dates and end dates below are estimated. The real dates will be provided in later development of the project.

|  |  |  |  |
| --- | --- | --- | --- |
| **Phasing** | **Effort** | **Start date** | **Finish date** |
| 1. Sprint 1 | Low | 04.09.2023 | 22.09.2023 |
| 1. Sprint 2 | Medium | 23.09.2023 | 13.10.2023 |
| 1. Sprint 3 | Medium | 14.10.2023 | 10.11.2023 |
| 1. Sprint 4 | High | 11.11.2023 | 01.12.2023 |
| 1. Sprint 5 | Medium | 02.12.2023 | 22.12.2023 |
| 1. Sprint 6 | High | 23.12.2023 | 19.01.2024 |

# Testing strategy and configuration management

## Testing strategy

Mainly, the testing will be conducted with unit tests. As for RESTful APIs, they will be tested using Postman. The testing will ensure that the code is working as intended for optimal user experience. In order to stay away from incorrect or dummy data in the main repository, a mock repository will be included in the testing of the code and the logic so that the main repository is not affected in any way.

Integration testing, end-to-end testing, acceptance testing and frontend testing will also be considered to test different components of the application.

Moreover, Mockito will be used to create mock objects and not affect the primary database. The focus will be on testing the functionality of the business logic and avoid testing the functionality of the database.

## Test environment and required resources

During implementation, a CI/CD environment will be used in order to test any published builds in the Git repository. This will ensure that no flawed methods and functions have been left unfixed and will improve quality of the application. TestCherry will also be used in testing to save valuable time of implementing the product.

## Configuration management

For configuration management and version management, we will be using a GitLab repository. A link will be provided to the stakeholders so that they have access to the latest or earlier versions of the product at all times. This will also ensure that in case of breaking the application, an earlier version will be available and less progress will be lost.

# Risk and mitigation

## Risk and mitigation

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Risk** | **Impart** | **Probability** | **Prevention activities** | **Mitigation activities** |
| 1. The project may get deleted or lost | Huge | Low | Every project version will be available to download from GitLab | In case of the project getting lost, it can be downloaded again from GitLab |
| 1. Code may be lost or deleted | Huge | Medium | Every project version will be available to download from GitLab | In case of code getting lost, it can be downloaded again from GitLab |
| 1. The application may crash or restart, leading to loss of work | Huge | High | The project will constantly be saved in order to not lose work | In case the application crashes, the file will already be saved and no major changes will be lost |
| 1. The developer may experience a burnout | Huge | Medium | The developer will have a perfect balance between working time and rest | The developer will improve their focus in order to complete work efficiently and have more time to “cool down” |
| 1. Miscommunication may occur between the developer and the stakeholders | Huge | Medium | All of the stakeholders’ requirements will be written down | The requirements will be noted down in a short span of time after a discussion is made in order to not forget important details, features, or ideas |
| 1. The stakeholders may be absent when needed | Medium | Medium | Feedback from the stakeholders will be asked for constantly so that project development is not slowed down in case of stakeholders’ absence | Stakeholders can be contacted via email or Teams |