Text

Description automatically generated

Qatar University

CMPS – 310 Software Engineering - L01- B03 Fall 2021

Course Project Milestone 2

**Instructor:** Alaa Hussein

|  |  |
| --- | --- |
| **Students name & ID:** |  |

**Student Name:** Elbaraa Elhawary  **QUID:** 201703540 **Effort Given:** 25%

**Student Name:** Ezeldin Ahmed **QUID:** 201803302 **Effort Given:** 25%

**Student Name:** Mahmoud Talkhan **QUID:** 201908144  **Effort Given:** 25%

**Student Name:** Mohammed Al-Qeraisi **QUID:** 201909907 **Effort Given:** 25%

**DECLARATION**: We hereby certify that no part of this project or product has been copied from any other student’s work or from any other sources except where due acknowledgement is made in the project. No part of this project/product has been written/produced for us by any other persons.

**Submit Date:** 14 – 11 – 2021

Table of Contents

[Task 1 (Constraints and NFRs) 3](#_Toc85525348)

[A) Constraints 3](#_Toc85525349)

[B) Quality Attributes (NFRs) 3](#_Toc85525350)

[Task 2 (Software Architecture) 3](#_Toc85525351)

[A) Proposed Architecture 3](#_Toc85525352)

[B) Explanation 3](#_Toc85525353)

[Task 3 (Software Implementing and Screen Shots) 3](#_Toc85525354)

[A) Implementation 3](#_Toc85525355)

[B) Screen Shots 3](#_Toc85525356)

[Issue Visit Visa use case 3](#_Toc85525357)

[Task 4 (Testing) 3](#_Toc85525358)

[Test Case & result 1: 3](#_Toc85525359)

[Explanation 3](#_Toc85525360)

[Task 5 (Last two NFRs Testing) 3](#_Toc85525361)

# Task 1 (Constraints and NFRs)

## Constraints

|  |  |
| --- | --- |
| Development Constraints | * The interface of the system must keep the same if the functions change. * The system must allow adding new features and functions to the system in the future so the system must be modifiable. |
| Technical Constraints | * The system must run on multiple systems Linux, iOS, Unix, etc. * The system must handle growth of 150% of users within 1 year. * The system must be protected from the direct use by the user. |
| Resource Constraints | * 2 servers. * 5 technical staff. |
| Practical Constraints | * Customer privacy must be respected. |
| Business Constraints | * The system must run as plug-in for the registration system that uses Google Firebase. |
| Schedule Constraints | * The system must be delivered within 3 weeks. |

## B) Quality Attributes (NFRs)

|  |  |
| --- | --- |
| Quality Attribute | Architecture Requirement |
| Portability(see 1.) | The system can run on different operating systems Linux, iOS, Unix, etc. |
| Performance | The Application performance must handle 100 concurrent member’s requests for the time being. |
| Availability(see 2.) | The system must be available during the weekdays. |
| Modifiability | More functions can be added or removed later without changing the interface. |
| Usability(see 3.) | The user interface of the system must be user-friendly. |
| Security | The payment information of members must be authenticated, and confidential, and only authorized people can access it. |

(1.) We assumed this quality attribute because we used java programming language since Java Virtual Machine (JVM) is cross platform and achieves the same end results independent of the operating system so, our application can run on different operating systems.

(2.) We assumed this quality attribute because the if the system is not available for the customers during the working hours, no applicant will be able to submit his request etc.

(3.) We assumed this quality attribute because we need our system to be used easily by any user’s type.

# Task 2 (Software Architecture)

## A) Proposed Architecture

According to our quality attributes and constraints we decided to choose " Object-Oriented System" to be our proposed architecture of Call-and-return style.

## B) Explanation

For the Portability, Programs made using Java OOP will run on different operating systems Linux, iOS, Unix, etc. without any modifications. For Usability, this architecture supports the separation of concerns so it will be useful for our system. For Performance, Object-Oriented Systems tend to be slow. The desired speeds may not be easy to achieve using this style but in our system, we used three layers, which are the MVC, these layers are separated so any modification on any layer won’t affect the other layers.

For Modifiability adding and updating functionalities is possible, for instance, adding and modifying different payment options without causing major changes.

For Security data is private, it cannot be accessed directly from the views, the controller is used to "access" the private data in the model.

# Task 3 (Software Implementing and Screen Shots)

## A) Implementation

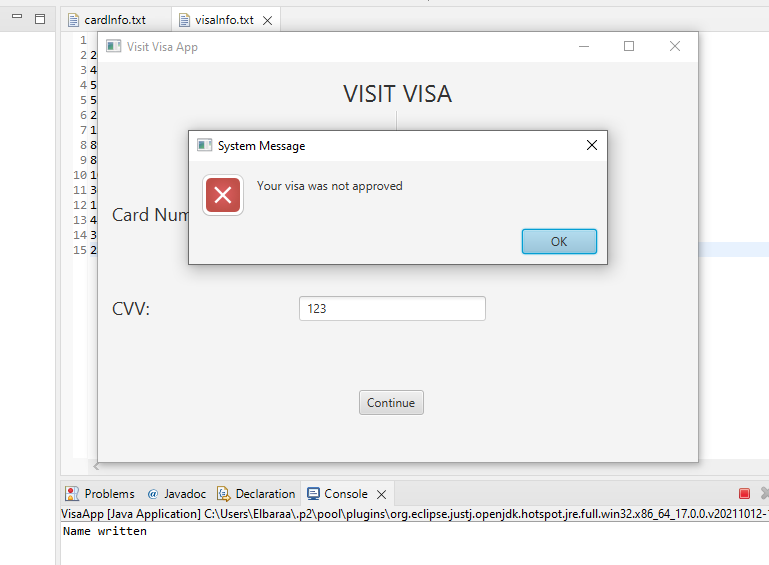
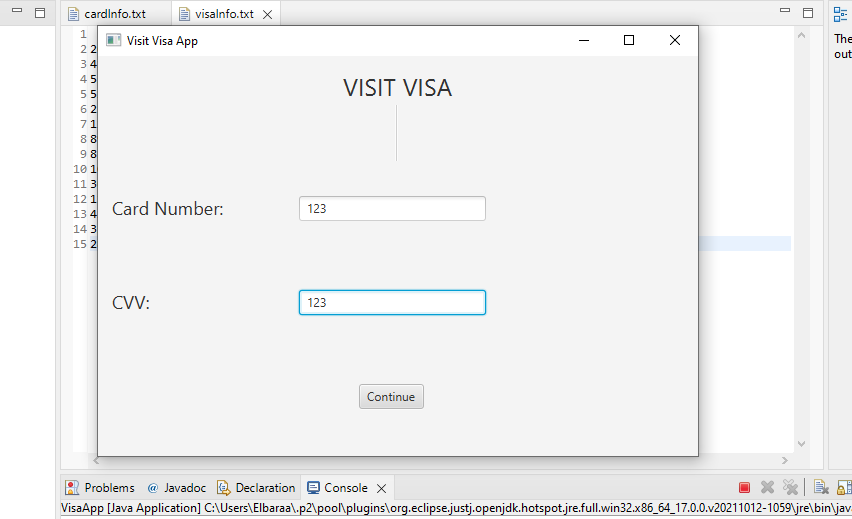
The use case required to be Implemented are done using Java language, and the source code is attached.

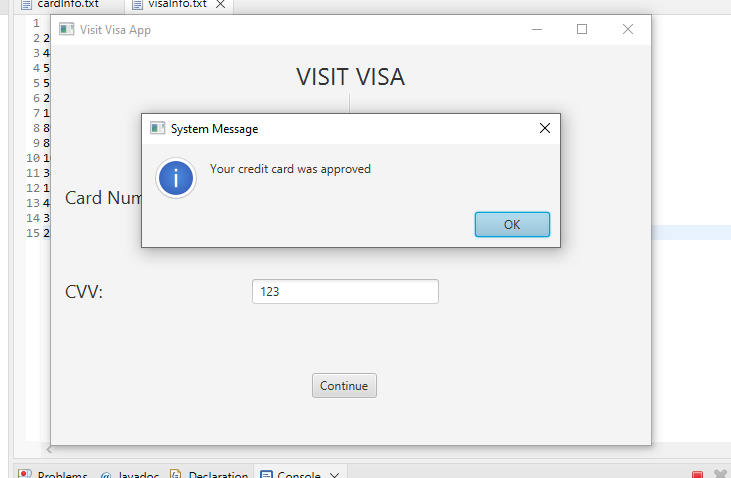
## B) Screen Shots

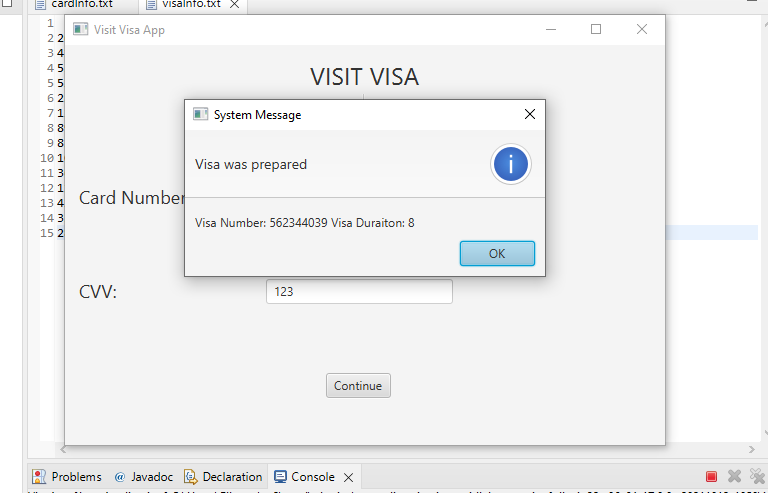
### Issue Visit Visa use case

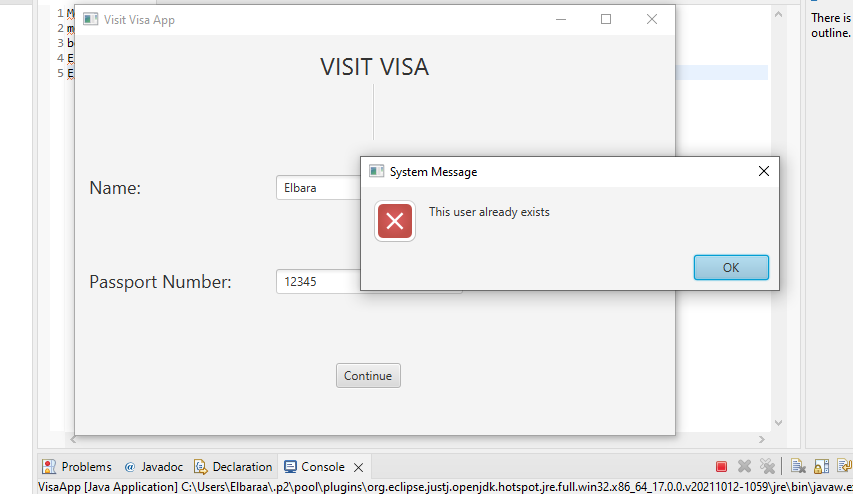
Graphical user interface, application

Description automatically generated









# Task 4 (Testing)

## Test Case & result 1:

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

## Explanation

We utilized the Junit library for testing since manually proving all of a program's features becomes time-consuming as the product grows larger. We tested our implementation by creating some clone objects of the class that we will be primarily operating on for our use case. As a result, our primary focus was to ensure accuracy within our use case, which we can confirm if our dummy objects produce the intended result.

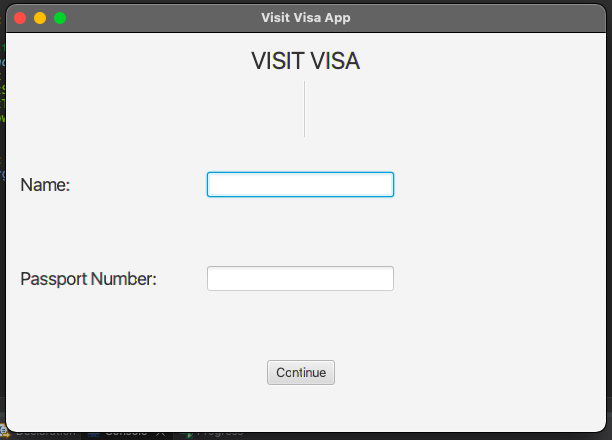
# Task 5 (Last two NFRs Testing)

To test the quality requirements, we have tested 2 NFRs, and they are as follow:

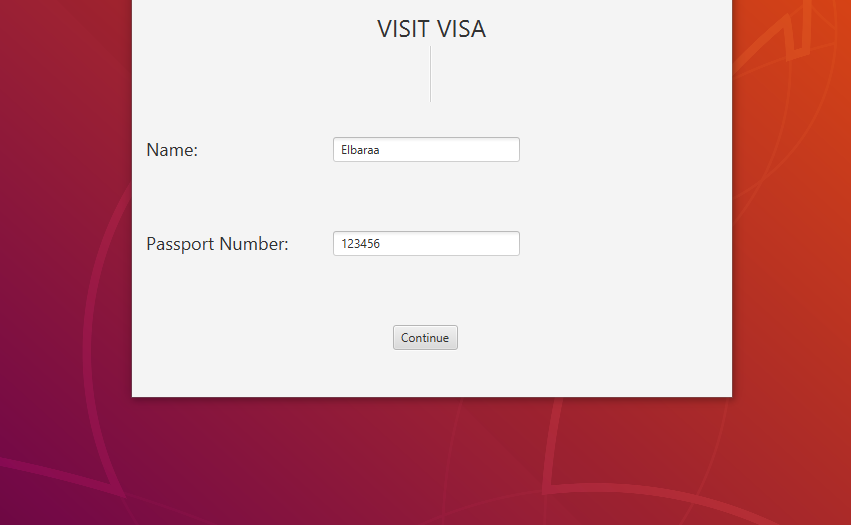
1- Portability:

We have tested our system on different operating systems, and it successfully worked on all of them.

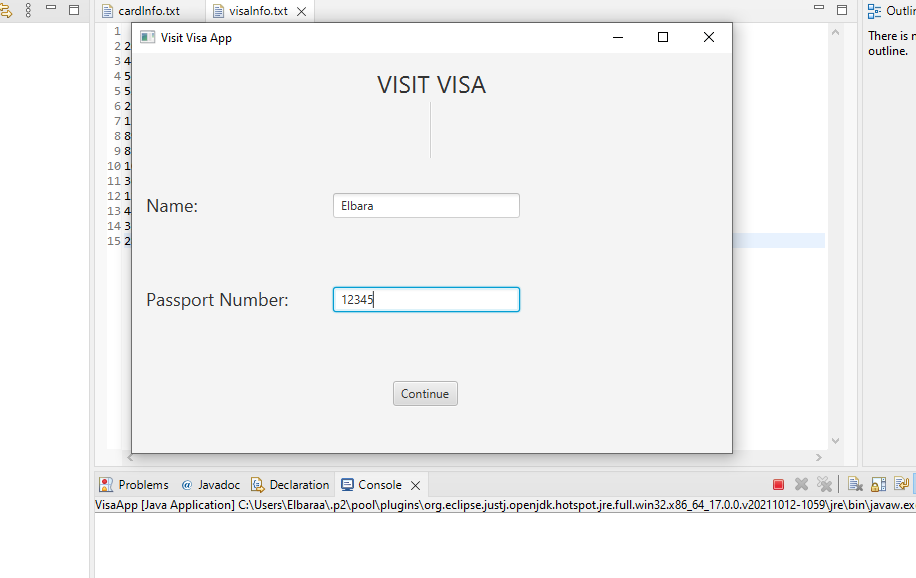
First, on Mac.



Second, Linux.



Last, Windows.



2- Usability:

We have made a user-friendly interface, so any person can use our system regardless of his/her technical background. For testing this part, we made some users test our system and we made a survey to provide feedback.

Chart, pie chart

Description automatically generated

Chart, pie chart

Description automatically generated

Chart, pie chart

Description automatically generated

Chart, pie chart

Description automatically generated