CS5002NI Software Engineering

DREAM TOURS & TRAVEL

45% Individual Coursework

2020-21 Spring

Student Name: MD OSHAMA

London Met ID:19031128

College ID:NP01CP4A190177

Coursework given:25th may

Submission Date:26th may

I confirm that I understand my coursework needs to be submitted online via Google Classroom under the relevant module page before the deadline in order for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a mark of zero will be awarded.

Table of content

Contents

List of figure

List of Tables

SECTION A

A.1

Software engineers are computer science professionals who use knowledge of engineering principles and programming languages to build software products, develop computer games, and run network control systems.

Professional responsibilities of Software engineer

Talking about Professional responsibilities of a software engineer

1. Improving system quality by identifying issues and common patterns, and developing standard operating procedures
2. Enhancing applications by identifying opportunities for improvement, making recommendations and designing and implementing systems
Maintaining and improving existing codebases and peer review code changes
3. Liaising with colleagues to implement technical designs
4. Investigating and using new technologies where relevant
5. Providing written knowledge transfer material

Importance of Good software engineer

A good software engineer is someone who is not only competent at writing code, but also competent in everything else required to build, deliver, and ship valuable software. A good software engineer is someone that their company can trust to help them make money.
Some people might disagree holistically, or at the margins, with this. They might emphasize things like algorithms and data structures knowledge, mathematical aptitude, or prodigious command of programming languages.
Qualities
1. Knowledge of the full stack
2. Ability to learn, adapt and grow
3. Business acumen
4. Time management
5. Interpersonal skills

So if there is no presence of a good software engineer the company will lack the features and might face a lot of problems.

Key challenges faced in the field of software engineering

Rapid technology advancement

Every technology advancement is a blessing for the IT industry. But at the same time, technology evolving at a phenomenal rate leads to an added pressure for software development professionals to leverage these upcoming technology trends in software product development to gain a cutting edge over competitors and stand out in the market.

Increasing customer demands

Software projects are generally conceptual and are aimed at designing and developing software products that meet varied customer demands. To develop even the simplest application or product, developers must clearly understand the underlying business concept and bring in the required features to satisfy the growing customer demands.

Time limitations

Software development is a time-game. Developers work under pressured environments and strive to complete project requirements within strict and scanty timelines. This is especially a challenge when working with international clients on multiple time-zones. Time constraints often bring down efficiencies of development teams and lead to mediocre quality software products in the end.

Limited infrastructure/resources

Another challenge faced by majority of software development companies is a lack of resources or IT infrastructure to execute projects effectively. This could mean a lack of high-performance software development tools, powerful computing platforms, inefficient data storage architectures or improper networks and connectivity. Such hindrances bring down productivity and performance of software development teams and impact the overall result.

Conflicts with software testing teams

In a classic software development project, interpersonal conflicts occur inevitably between software development and testing teams. Several factors contribute to such conflicts like working under high performance pressure, different mindsets, difference in job roles and the very opposite nature of development and testing. If not controlled and managed effectively, these conflicts could hamper the overall project adversely.

Requirements Elicitation

It’s a process of interacting with customers and end-users to find out about the domain requirements, what services the system should provide, and the other constrains.

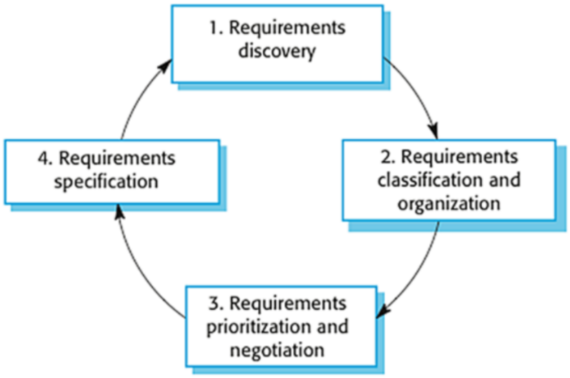
Domain requirements reflect the environment in which the system operates so, when we talk about an application domain we mean environments such as train operation, medical records, e-commerce etc.

It may also involve a different kind of stockholders; end-users, managers, system engineers, test engineers, maintenance engineers, etc.

We typically start by gathering the requirements, this could be done through a general discussion or interviews with your stakeholders, also it may involve some graphical notation.

Then you organize the related requirements into sub components and prioritize them, and finally, you refine them by removing any ambiguous requirements that may raise from some conflicts.

Process


1. Requirements discovery
2. Requirements classification and organization
3. Requirements prioritization and negotiation
4. Requirements specification

The process of requirements elicitation and analysis

We typically start by gathering the requirements, this could be done through a general discussion or interviews with your stakeholders, also it may involve some graphical notation.

Then you organize the related requirements into sub components and prioritize them, and finally, you refine them by removing any ambiguous requirements that may raise from some conflicts.

A.3

Goals of Testing

1. Quality
2. Customer Satisfaction
3. Risk management
4. Reduced maintenance cost
5. Improved software testing process
6. Bug Prevention

Difference Between Black Box Grey Box and White Box Testing

S.no	Black Box	White Box	Grey Box
1	Knowledge of internal working structure (Coding of software) is necessarily required for this type of testing. Knowledge of internal working structure (Code) is not required for this type of testing. Only GUI (Graphical User Interface) is required for test cases.	Knowledge of internal working structure (Coding of software) is necessarily required for this type of testing.	Partially Knowledge of the internal working structure is required. Partially Knowledge of the internal working structure is required.
2.	Black Box Testing is also known as functional testing, data-driven testing, and closed box testing	White Box Testing is also known as structural testing, clear box testing, code-based testing, and transparent testing.	Grey Box Testing is also known as translucent testing as the tester has limited knowledge of coding.
3	The approach towards testing includes trial techniques and error guessing method because tester does not need knowledge of internal coding of the software.	White Box Testing is proceeded by verifying the system boundaries and data domains inherent in the software as there is no lack of internal coding knowledge.	If the tester has knowledge of coding, then it is proceeded by validating data domains and internal system boundaries of the software.
4	The testing space of tables for inputs (inputs to be used for creating test cases) is pretty huge and largest among all testing spaces.	The testing space of tables for inputs (inputs to be used for creating test cases) is less as compared to Black Box testing.	The testing space of tables for inputs (inputs to be used for creating test cases) is smaller than Black Box and White Box testing.
5	It is very difficult to discover hidden errors of the software because errors can be due to internal working which is unknown for Black Box testing.	It is simple to discover hidden errors because it can be due to internal working which is deeply explored in White Box testing.	Difficult to discover the hidden error. Might be found in user level testing.

Table 1: Difference between black box, white box and grey box testing

Regression Testing

Regression testing is a software testing practice that ensures an application still functions as expected after any code changes, updates, or improvements. Regression testing is responsible for the overall stability and functionality of the existing features. Whenever a new modification is added to the code, regression testing is applied to guarantee that after each update, the system stays sustainable under continuous improvements.

When to use regression testing

1. A new requirement is added to an existing feature
2. A new feature or functionality is added
3. The codebase is fixed to solve defects
4. The source code is optimized to improve performance
5. Patch fixes are added
6. Changes in configuration

Verification

Verification is the process of checking that a software achieves its goal without any bugs. It is the process to ensure whether the product that is developed is right or not. It verifies whether the developed product fulfills the requirements that we have. Verification is static testing.
Verification means Are we building the product right?

Validation

Validation is the process of checking whether the software product is up to the mark or in other words product has high level requirements. It is the process of checking the validation of product i.e. it checks what we are developing is the right product. it is validation of actual and expected product. Validation is the dynamic testing.

Differentiating validation and verification

Verification	Validation
Code inspection	‘White Box’ Module Testing
Module design review	‘Black Box’ Module Testing
Requirement reviews	Integration testing
	Acceptance testing

Section B

B.1.a

Use Case Diagram

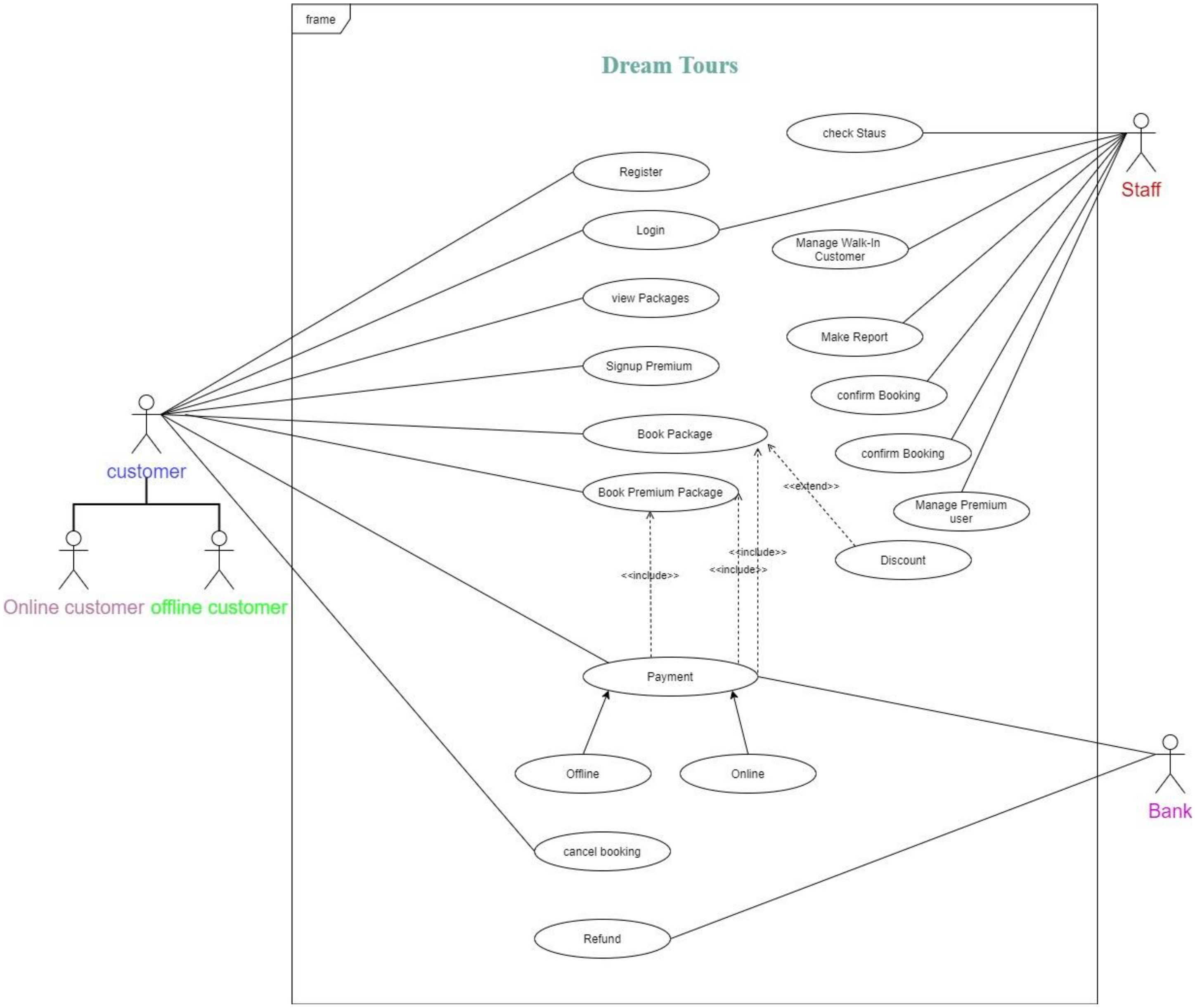


Figure 1: Use Case Diagram

B.1.b

Manage Premium User

High Level Description

Use Case	Manage Premium User
Actor	Staff
Description	Staff Manages the Premium User either offline or online premium bookings are handled by the staff.

Table 2: High level Description Manage Premium user

Expanded Level Use Case Description

Use Case	Manage Premium User
Actor	Staff
Description	Staff Manages the Premium User either offline or online premium bookings are handled by the staff.

Customer	Staff
1. Customer Does the registration or signup to Premium user	2. staff checks the form and confirms the Premium Membership.
3. Customer asks for the premium packages present	4. Staff or system displays the Premium packages present for the booking.
5. Customer views the packages and ask for the booking.	6. Booking in process, ask for the Payment
7. Does the Payment	8. Confirms the payment
	9. Confirm Booking
10. Check out	

Table 3: Expanded Description Manage Premium user

B.1.c

Manage Walk In Customer

High Level Description

Use Case	Manage Walk in Customer
Actor	Staff
Description	Staff manages the customer who visit physically for the package either normal package or premium Package.

Table 4: High Level Description Walk In customer

Expanded Level Description

Use Case	Manage Walk in Customer	
Actor	Staff	
Description	Staff manages the customer who visit physically for the package either normal package or premium Package.	
Customer		Staff
1. Customer enter the details in registration form		2. staff checks the details and confirms the registration process
3. Customer asks to show the available packages		4. staff shows different packages including the premium packages.
5. Customer select a package		6. Staff checks the status of package and ask for payment
7. Customer give a deposit or does full payment		8. Staff confirms the payment and confirm booking
9. Checkout		

Table 5 : Expanded Description Walk In Customer

Activity Diagram

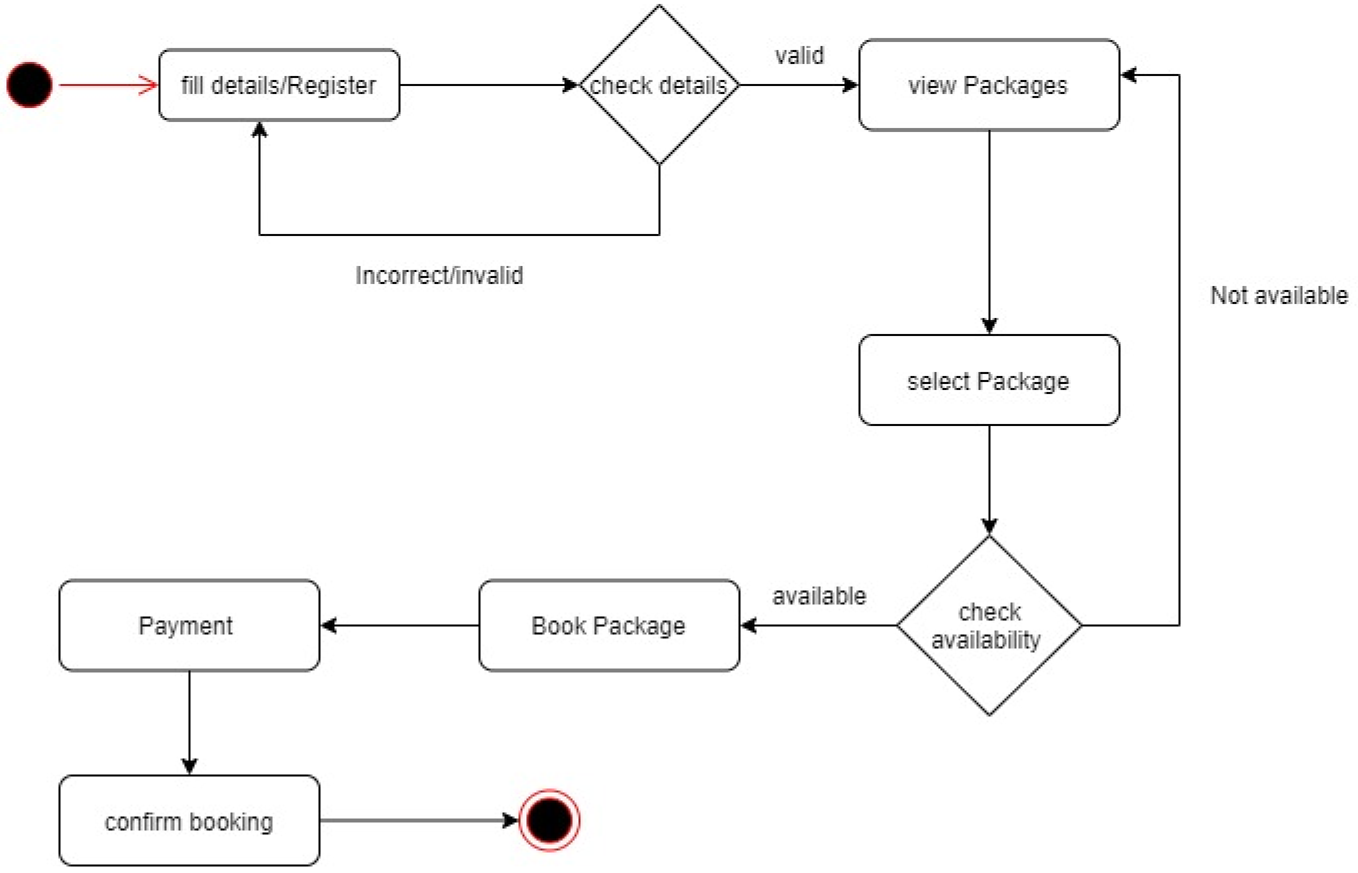


Figure 2: Activity Diagram

B.3
Class Diagram

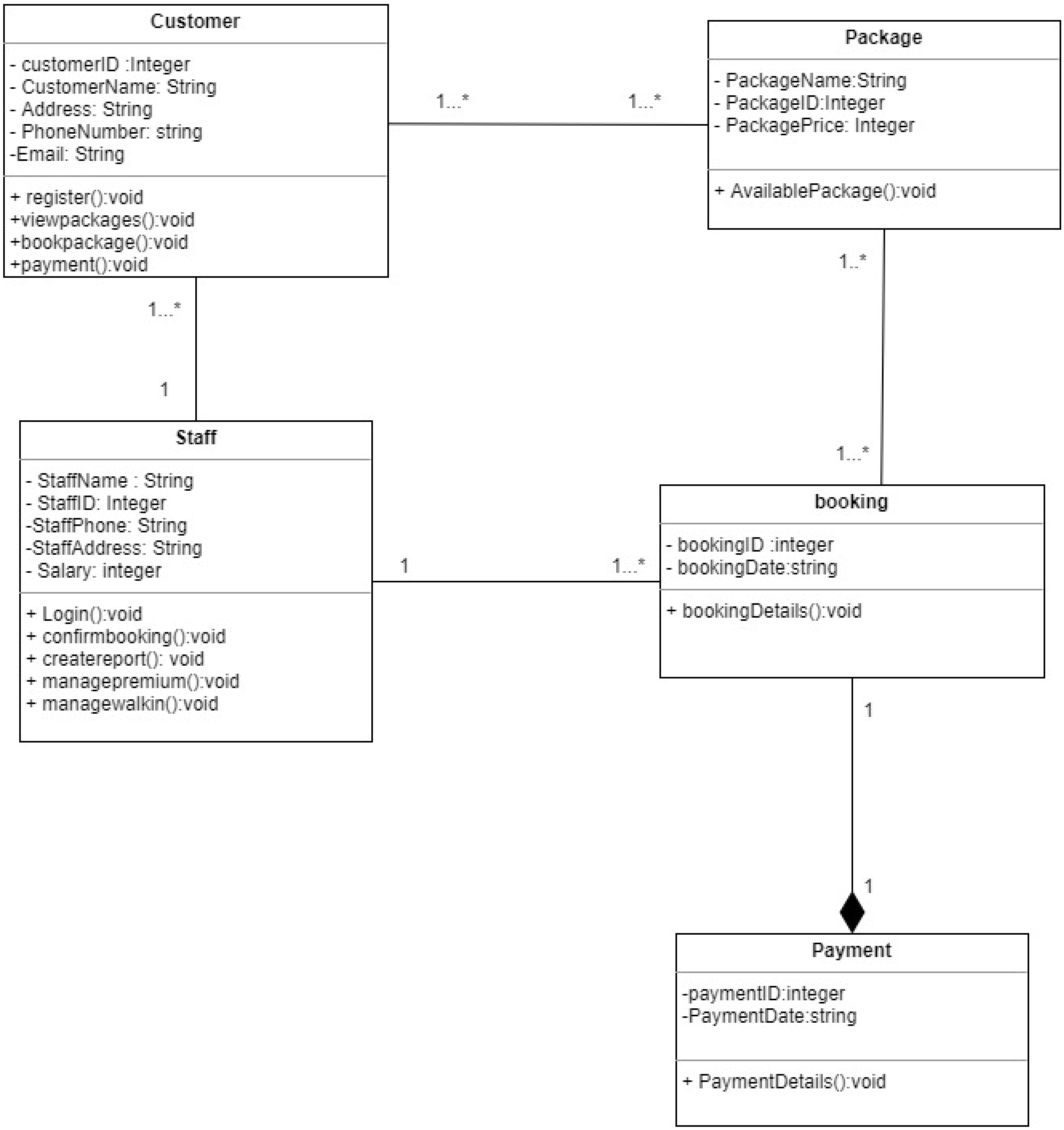


Figure 3: Class Diagram

Thank You