

BlueCollar

Handyman Application

Software Engineering Project
BINF III A

Content

01

Introduction

- 1.1 Problem Description & Solution
- 1.2 Aim and Objectives
- 1.3 Description of the BlueCollar App

03

Software Design & Modeling

- 3.1 Use Case Diagrams
- 3.2 Activity Diagrams
- 3.3 Sequence Diagrams

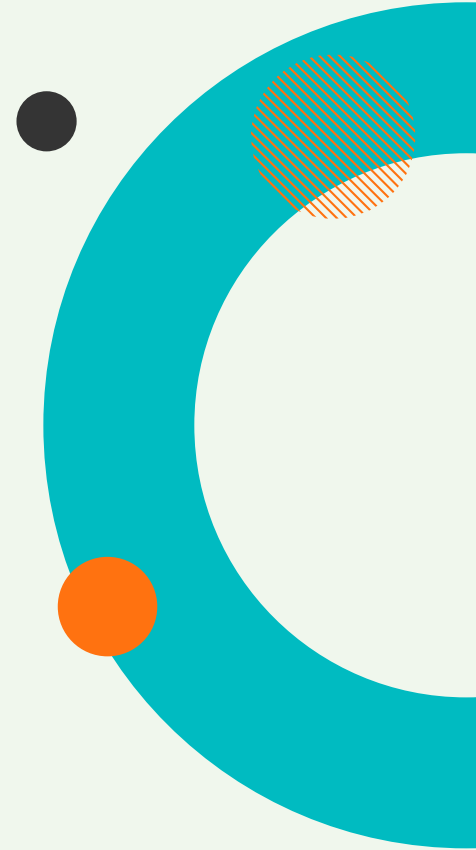
02

Requirements Engineering

- 2.1 Requirements Elicitation & Analysis
- 2.2 Requirements Specification
- 2.3 Requirements Validation
- 2.4 Requirements Management

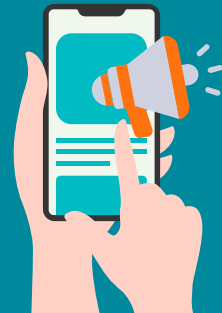
04

Evolution and Maintenance





01 Introduction



PROBLEM DESCRIPTION and SOLUTION

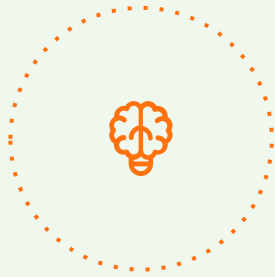
By doing a thorough research of challenges in our daily life we came up to the conclusion that one problem that has always been an obstacle was finding a handyperson in the in the moment without having to ask our friends, neighbors, or relatives for a "known guy" that would offer us help.

Keeping this in mind we thought that a **Handyman Service Web App** would be the ideal solution to offer these services with only a click away.



AIM AND OBJECTIVES

The aim and objectives of this project are to create a website that fulfills both customer and service providers' (handymen) *needs* and *requirements*.



Aim

The application should be:

- ✓ well designed
- ✓ functional
- ✓ user-friendly
- ✓ have a responsive design.



Objectives

Our main objectives are:

- ✓ User Engagement
- ✓ Filter target audience
- ✓ Build a feasible application
- ✓ User-friendly application
- ✓ Keep it simple!

BlueCollar - Product Description

BlueCollar application is built and designed as a platform that brings together customers in need of service and service providers of a variety of expertise. A handyman is a person skilled at a wide range of repairs, typically for keeping buildings, shops, or equipment around the home in good repair. This includes services of a wide variety starting from electric appliances to house repair problems.

The application is **web-based**, necessitates an internet connection, and works with most web browsers and operating systems. Initially, it is intended to find use in the Albanian region.

The application will be built using HTML, CSS, JavaScript, PHP, and MySQL for database connection needs. As seen fit, the appearance design will be easy on the eye and response time will be adequate. The application aims for simplicity and ease of use.

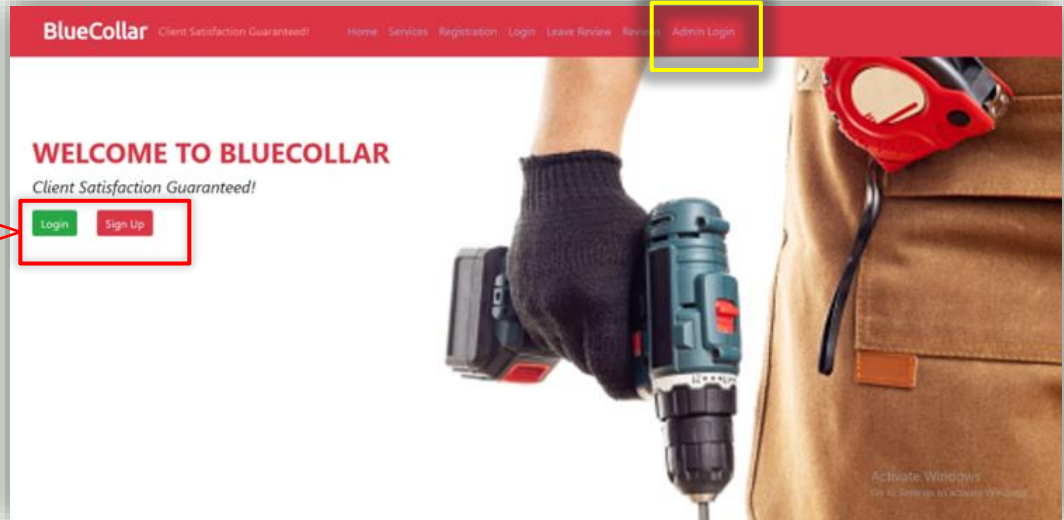


BlueCollar - Product Description

The application first displays a greeting and prompts a **log-in** and **sign-up** form.
When signing up you need to specify your profile as either a **customer** or **handyman**.

CUSTOMER

HANDYMAN







BlueCollar


Customer Profile Description

Create an Account

 **Name**

 **Email**

We'll never share your email with anyone else.

 **New Password**


Sign Up

Note - By clicking Sign Up, you agree to our Terms, Data Policy and Cookie Policy.


The customer signs up by entering their personal information such as their **first name**, **surname**, **email**, **contact information**, **payment details**, and **address**.

Once the customer has *signed up* and *logged in*, he/she is able to utilize the functionalities of the application. The user will be displayed a list of handymen with their respective information. The user is able to schedule the services of the desired handyman and make the payment online.

Customer Log In

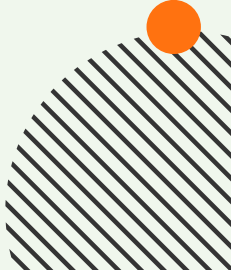
 **Email**

We'll never share your email with anyone else.

 **Password**

Login

Back to Home





BlueCollar

Handyman/ADMIN Profile Description

The handyman profile will require **name, surname, city, payment, contact, email, service area, working hours, and pay rate** information.

When logged in as a handyman, you will be displayed your appointments including the payment status. You may as well edit your information details as needed. Access to these respective functionalities will be through buttons and information will usually be displayed in a table format, entering data and editing will be done using forms.

Administrator Log In (Only Authorized Management)

Email

We'll never share your email with anyone else.

Password

[Login](#)

[Back to Home](#)

BlueCollar

Dashboard

Work Order

Requests

Technician

Requester

Work Report

Reviews

Change Password

Logout

Requests Received

61

View

Assigned Work

61

View


No. of Technician

1

View

List of Requesters

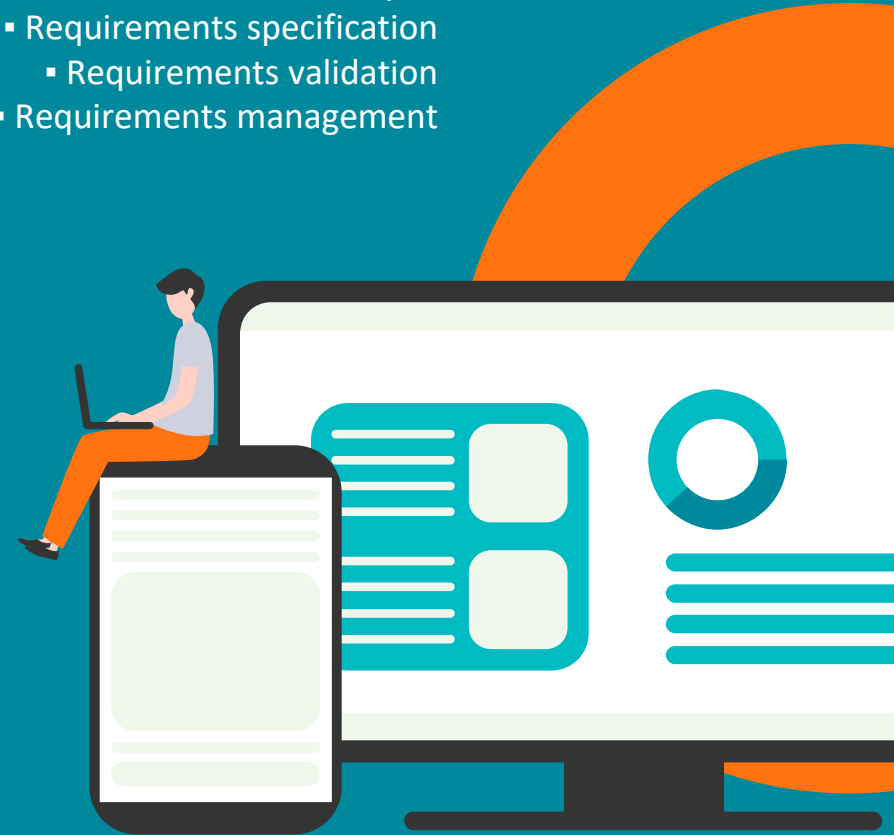
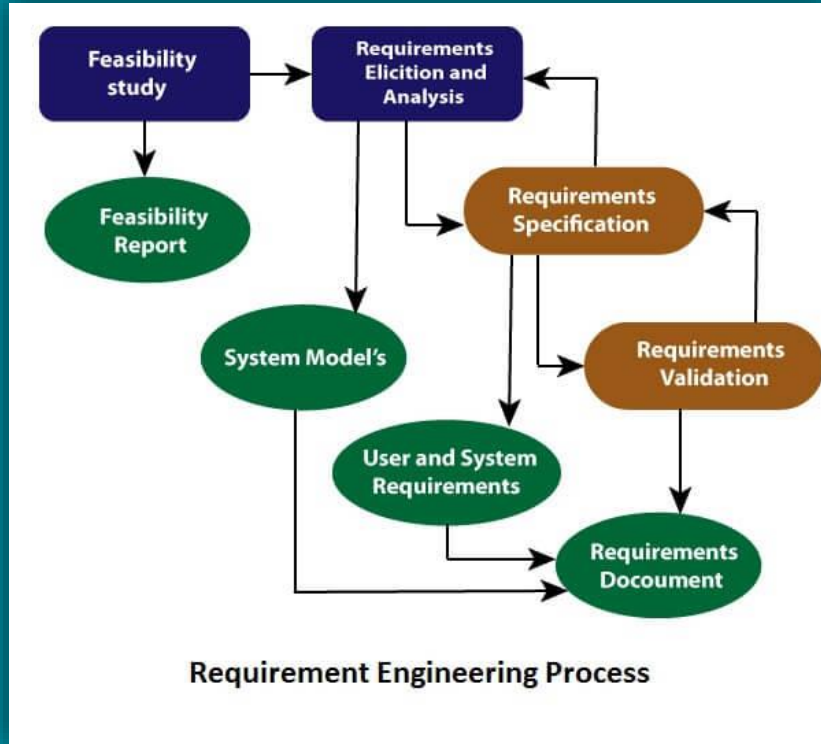
Requester ID	Name	Email
13	Alesia Giana	alesiagiana@gmail.com



02 Requirements Engineering

Requirements engineering process

- Requirements elicitation & analysis
 - Requirements specification
 - Requirements validation
- Requirements management



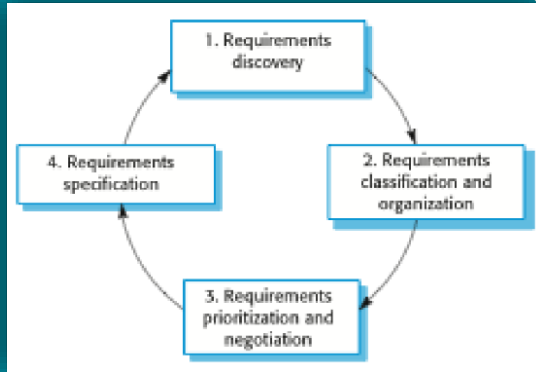
Requirements Elicitation and Analysis

Requirement Discovery

This process is known as “The process of **gathering information** about the required and existing systems and distilling the user and system requirements from this information.”

In order to discover the requirements we need to identify the **stakeholders**. The stakeholders can be divided into *internal* and *external* stakeholders.

We need to interact with stakeholders in order to acquire information that will help us distil the user and system requirements. This can be done through different methods and we will be using **interviewing**. The interview will consist of a mix of closed and open questions.



1. What features should the application provide for the handyman?

Stakeholder	Handyman	Engineers
1.	I would like for appointments at home to be made online and to be able to view my appointments.	The handyman should be able to edit any information that they enter
2.	I would like to cancel an appointment made or edit it if a personal problem came up for me.	
3.	I would like for the payment to be made online from the customer in order not to have any problems and to be more transparent.	
4.	I would like to post my details in a wall post, with the essential information to contact me, leave an appointment.	

- The handyman should be able to create a profile in the application.
- Be able to edit their profile information.
- Be able to list their service details for viewing.
- Be able to edit their service details.
- Be able to view appointments made, edit, decline them if wanted to do so.
- Be able to view payments made for the service.
- Be able to view the ratings with reviews.

Requirements Elicitation and Analysis

Requirements classification and organization

We can group the related requirements and organize them into coherent clusters according to how relevant they are to each other.

We also group and organize according to the clusters of profile user:

Handyman:

1. The application should allow the handyman to sign up and log in with the respective information.
2. The handyman should be able to post their information details.
3. The handyman should be able to view their appointments.
4. The handyman should be able to cancel, edit their appointments.
5. The handyman should be able to view the payments made to them.
6. The handyman should be able to view their rating and review.
7. The handyman should be able to edit their information posted.
8. The handyman should be able to view other handyman listings.
9. The handyman should be able to use tabs for the services to navigate for ease of use.

Appointments:

- ✓ The handyman should be able to view their appointments.
- ✓ The handyman should be able to cancel, edit their appointments.
- ✓ The clients should be able to make an appointment with a handyman.
- ✓ The clients should be able to view the appointments made.
- ✓ The client should be able to cancel, edit an appointment.

Prioritization and negotiation

In a total view of the initial requirements, we do not view conflicts between the requirements.

We can prioritize the initial requirements according to the clusters that we have depicted above, as such:

Sign up and log in
Handyman job posting
Appointment
Information detail
Rating and Reviews
Payment Requirements



Requirements Specification

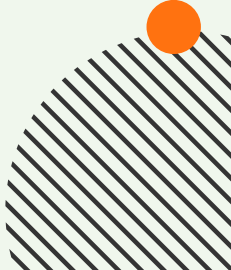
1. USER REQUIREMENTS

Statements in natural language plus diagrams of the services the system provides and its operational constraints. Written for customers.

2. SYSTEM REQUIREMENTS

A structured document that sets out detailed descriptions of the system's functions, services, and operational constraints. Defines what should be implemented so may be part of a contract between client and contractor.

The application should provide a sign up or login form for users (clients and taskers) in order for them to get the services that the app provides after creating an account or logging in to the existing one.

- a. The first page of the application should have the login and sign up button.
 - b. For every user who has already an account the data that he provides when logging in should match the one that is previously stored in the database.
 - c. For the new users the sign up window should be provided where they should fill in required information like name, last name, personal number or email address.
 - d. When clicking the login and sign up button the user should be redirected to a new window where a form created using php will appear.
 - e. When we create a new account the user will receive an email of configuration about the new account created.
- 

Functional Requirements

Functional requirements are requirements that the end user specifically requests as basic system facilities and describe ways a product must behave.

There are several types of functional requirements that can be part of the software system such as:

1. Operations and workflows the system must perform
2. Formats and validity of data to be input and output by the application
3. User interface behavior
4. Data integrity and security
5. Safety and regulatory measures
6. Validation of user access/authorization for use and modification (create/modify/delete) of the system.

Requirement Statement	Must/Want	Comments
1. System must allow new members (clients & handypersons) to sign up and login by providing to them all the needed data fields.	MUST	Form + data stored in DB
2. System must allow existing users to directly login with their saved credentials.	MUST	
3. The system must send a confirmation email when a <i>new</i> user account is created.	MUST	
4. The system must allow users to reset their password by clicking on " <i>Forgot Password?</i> ".	MUST	
5. The system must enable users to add, delete, or modify their data and information by clicking on " <i>EDIT</i> ".	MUST	



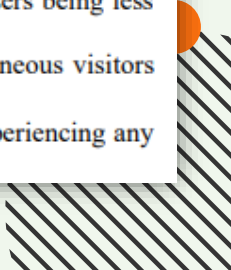
Non-functional Requirements

A non-functional requirement is a description of the nature of the object, its construction, or a constraint on its design or behavior. In short, these define how well it will function, including here aspects like performance, security, dependability, data accuracy, and so on.

These requirements must be built upon some ***standards***.

- ✓ *Usability:*
- ✓ *System Reliability, Maintainability, and Availability*
- ✓ *Scalability & Performance*
- ✓ *Security*
- ✓ *Localization*

NFR Documentation

1. It should take less than 1.5 seconds for the application's home screen to load.
 2. The landing page must respond in 6 seconds or less in a Chrome desktop browser, rendering text and images over an LTE connection included. This website must serve 800 users per hour.
 3. All website pages should load within 3 seconds with the total number of simultaneous users being less than or equal to 1500.
 4. The system must be scalable enough to handle, accommodate and support 8,000 simultaneous visitors while still maintaining an optimal performance.
 5. The system runs on Windows 10 and must be able to operate on Windows 11 without experiencing any performance or behavior changes.
- 

Domain Requirements

- ✓ **Privacy Policy & Data Protection:** The General Data Protection Regulation (GDPR) is a legal framework that sets guidelines for the collection and processing of personal information from individuals who live and outside of the European Union (EU).
- ✓ **Handyman's license:** The license is a necessary and mandatory document for verifying the employees' skills in the service sector they operate in.
- ✓ **NDA (Non-Disclosure Agreement):** is intended for protecting confidential information between parties, in this case between the organization and service providers.
- ✓ **Terms & Conditions:** TC is intended to protect the organization. They allow business owners to set their own rules (within the bounds of applicable law) for how their service or product may be used.
- ✓ **App Stores Requirements:** All requirements for app publishing established by Google and Apple guidelines must be met by mobile apps. They emphasize the importance of protecting personal data, such as health information and information obtained from minors, as well as intellectual property concerns regarding your mobile device.

Domain requirements reflect the environment in which the system operates. They are important because they often reflect the fundamentals of the application domain. If these requirements are not satisfied, it may be impossible to make the system work satisfactorily.

DOMAIN REQUIREMENTS OF BlueCollar Application


- ✓ Payment Gateway Regulations
- ✓ Privacy Policy & Data Protection
- ✓ Handyman's license
- ✓ NDA (Non-Disclosure Agreement)
- ✓ App Stores Requirements
- ✓ Terms & Conditions



Model Development

We have selected **the waterfall method** with a linear-sequential life cycle model to design our application. This model is really straightforward, simple to comprehend, and keeps the focus on the end goal at all times. There is no overlap between phases in a waterfall model; each step must be finished before the subsequent phase can start.

We chose this methodology since we are documenting our project in stages. Each phase is handled by thoroughly researching the features that must be included in the application and consulting with the client about what they need and want to see there. This methodology is more convenient for us to use because it enables us to focus solely on creating the software design with all of the features included and putting it into use after carefully going through each customer request and creating comprehensive requirement documents.





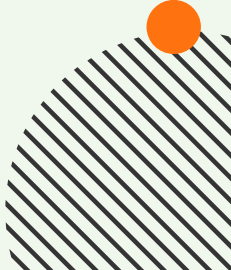
REQUIREMENTS VALIDATION

The main issue and challenge in developing an application is **“Are we building the right system?”** and the answer to this question can be given by validation.

Requirements validation is the process of confirming that the requirements we have given are appropriate for development and specifying the system that the client actually desires.

Requirements validation will assist us in spotting problems early on in the application development process and avert the need for unnecessary rework later on in the system development life cycle. Additionally, we will verify that the system contains all relevant requirements components and that the published requirements adhere to the expectations of stakeholders through requirements validation.

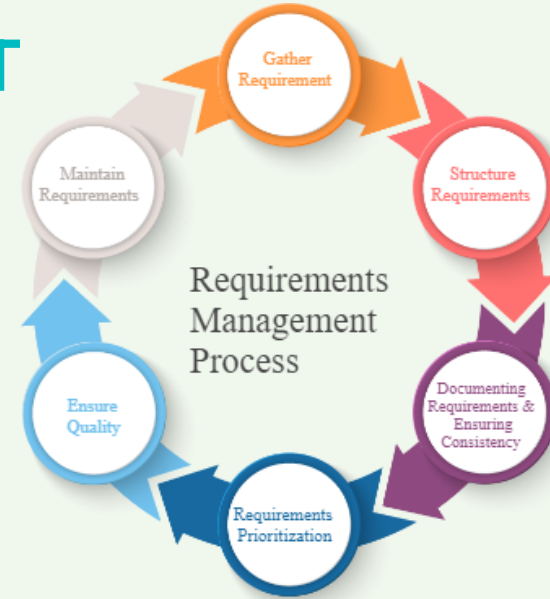
Requirements Validation Techniques

- 1) Review Checks
 - 2) Requirement reviews
- 

REQUIREMENTS MANAGEMENT

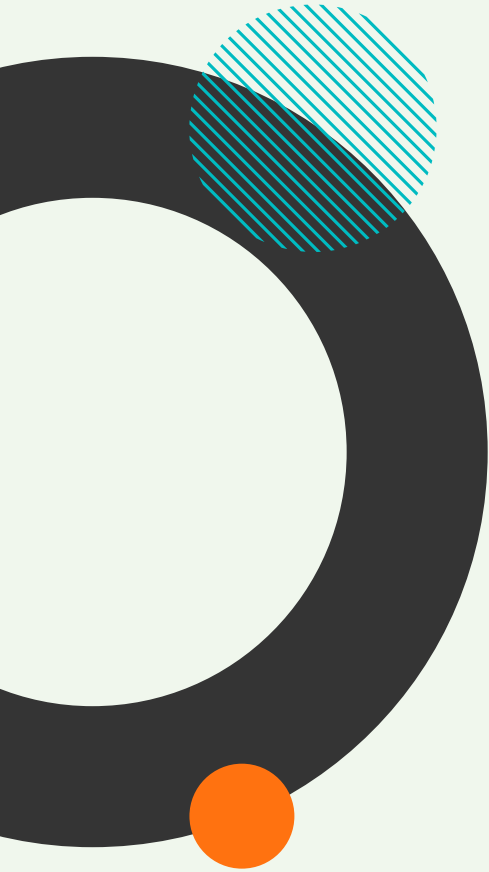
Requirements management is used to ensure that the objectives of product development are effectively attained.

A number of techniques have been developed for documenting, analyzing, prioritizing, and agreeing on requirements in order to guarantee that the engineering teams always have approved and up-to-date requirements. Requirements management provides a way to avoid mistakes by keeping track of requirements changes and encouraging communication with stakeholders throughout the project's engineering lifecycle.



Requirements management process

1. Collect initial requirements from stakeholders
2. Analyze requirements:
3. Define and record
4. Prioritize requirements
5. Agree on and approve requirements:
6. Trace requirements to work items



03

Software Design and Modeling



Design and Modeling Process

USE CASE DIAGRAM

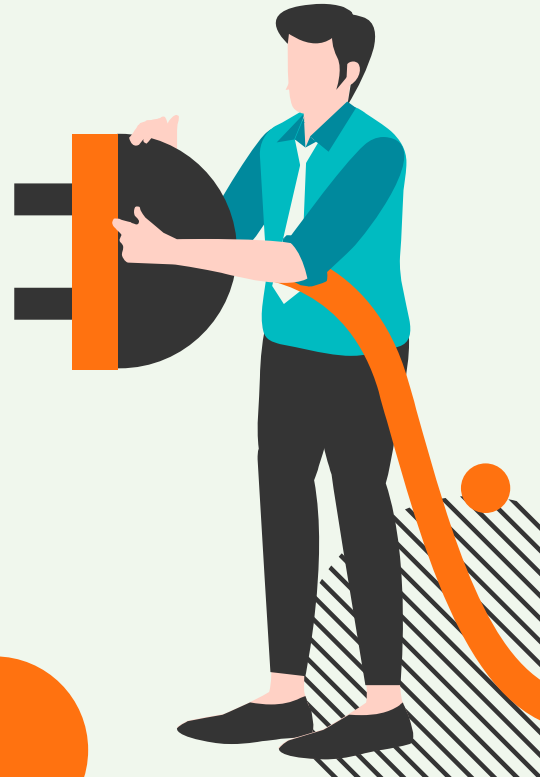
Display the interactions between the users and the system or other systems. It includes actors that interact with use cases, use cases themselves which are the functions, the communication link between them, and the system boundary to define what is inside and what is outside the system.

SEQUENCE DIAGRAM

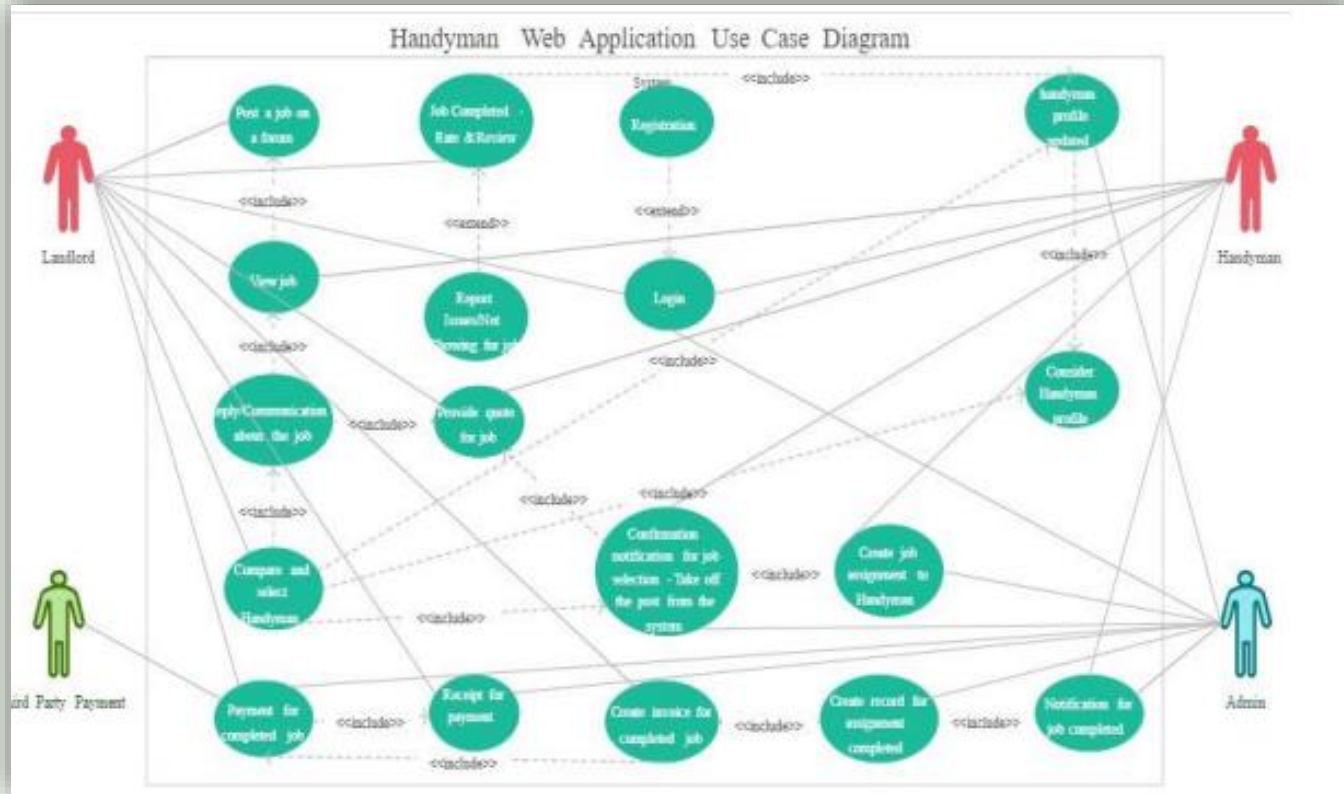
Demonstrates the sequence of interactions in the use case. The interaction between the objects is indicated by an arrow that shows a message and the dotted arrow a reply message, or synchronous and asynchronous. The vertical blocks from the lifelines or actors or objects indicate the time required, meanwhile, in some cases, we may have alternative blocks of what follows when an event occurs or not.

ACTIVITY DIAGRAM

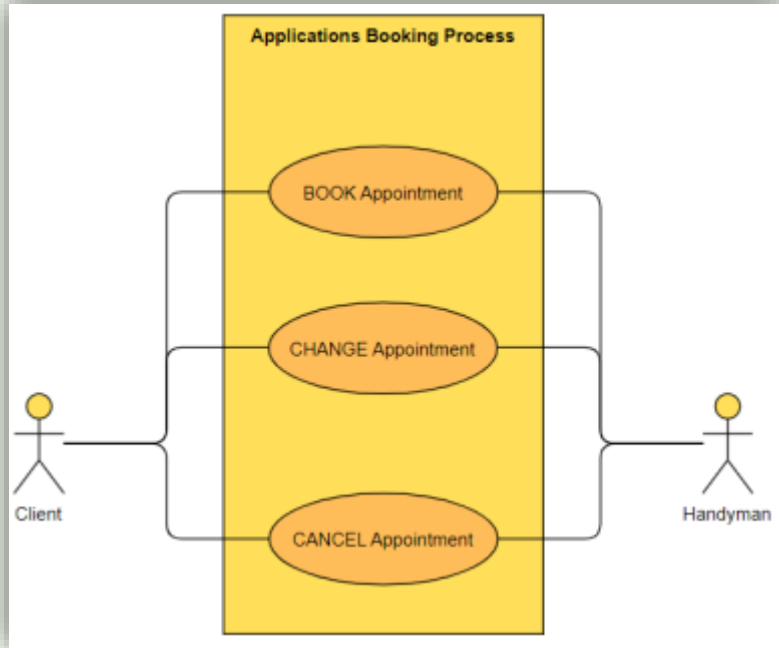
Through AD, we can comprehend a more complex flowchart in which the workflow is shown. We start with an initial state and continue the action flow with activities sometimes branching and sometimes merging, forks or guards used at times too, as seen fit while ending with an end state.



BlueCollar Use Case Diagram



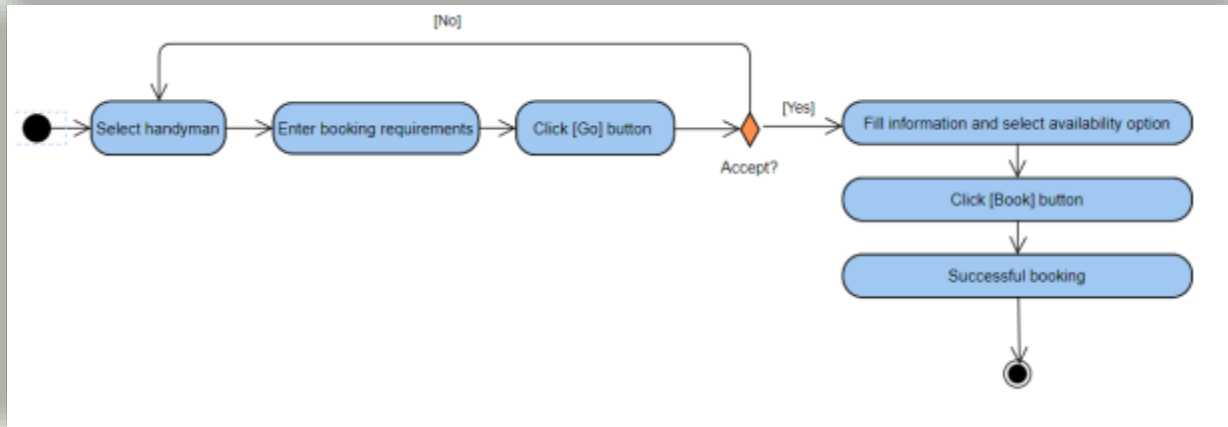
Appointment Use Case Diagram



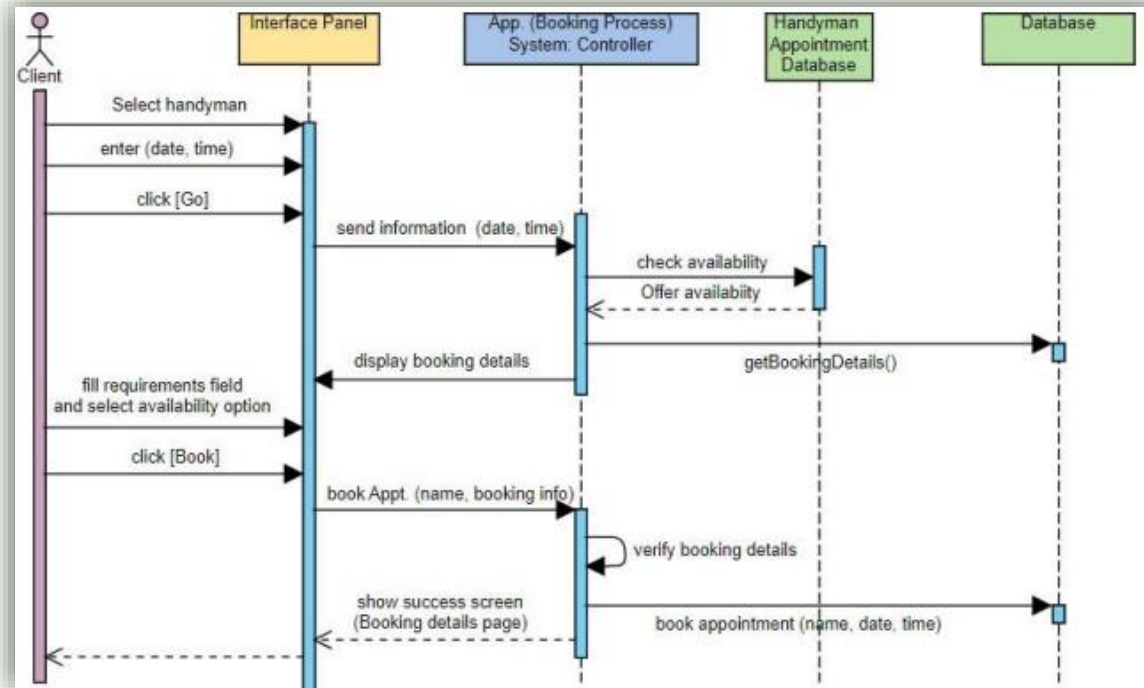
Tabular description

Handyman Application: Book Appointment	
Actors	Client, handyman, handyman application, app.DB
Description	Users must be able to book, cancel or change appointments.
Data	Users' information, appointment/booking information (date, time)
Stimulus	User command issued by the client by clicking on "Book" button
Response	Confirmation of the booking procedure
Comments	User must specify his/her information correctly in order to check availability and confirm the service appointment by the service provider.

Book Appointment *Activity Diagram*



Book Appointment *Sequence Diagram*



04 Evolution and Maintenance



Software Evolution and Maintenance



**Change identification
and evolution
processes**



**Handover problems
Bad Smells**



**Maintenance
Prediction**



**The emergency repair
process**



**Issues in business
value and system
quality assessment**



**Maintenance Cost
Factors**

Change identification and evolution processes

As we know from the literature, evolution is “the stage in a software system’s life cycle where it is in operational use and is evolving as new requirements are proposed and implemented in the system.”



The screenshot shows a web application interface for 'BlueCollar'. On the left is a sidebar menu with options: Dashboard, Work Order, Requests, Technician (highlighted), Requester, Work Report, Reviews, Change Password, and Logout. The main content area is titled 'Update Technician Details' and contains a form with the following fields: Emp ID (14), Name (John Doe), City (Tirana), Mobile (68795858), and Email (johndoe@gmail.com). At the bottom of the form are two buttons: 'Update' (red) and 'Close' (grey).

The change identification and evolution processes follow this cycle graph in the simplest terms containing 4 main stages:



Maintenance Prediction

Maintenance prediction is the process of predicting when maintenance tasks will need to be performed on a website in order to maintain its ideal functioning, security, and performance. It seeks to foresee faults or future maintenance requirements and proactively identify them before they result in major downtime or delays. By anticipating maintenance needs, website administrators can schedule and organize maintenance tasks in advance, minimizing the impact on users and the possibility of unforeseen issues.

In order to make a prediction we need certain data and information related to the website that we are building like data collection & analysis, metrics classification, continuous monitoring, and predictive models.



Maintenance Prediction



Predicting maintainability

What parts of the system will be the most expensive to maintain?

Since this website is a Custom-built one that is developed from scratch, it can be more costly in general to maintain. The most expensive part of our system would be protecting all the data. It is crucial for us to invest highly in making sure that all this data is protected when shared and stored.

Predicting maintenance costs

What will be the lifetime maintenance costs of this system? What will be the costs of maintaining this system over the next year?

Cost forecasting is challenging since it is reliant on several situations and variables. This cost will never be zero, as technology is continually developing.


Predicting system changes

What part of the system is most likely to be affected by change requests?

The parts of the system that are predicted to change more frequently are profiles of users and handymen, and appointments. We have left it in their hands to change personal information as many times as they need.

How many change requests can be expected?

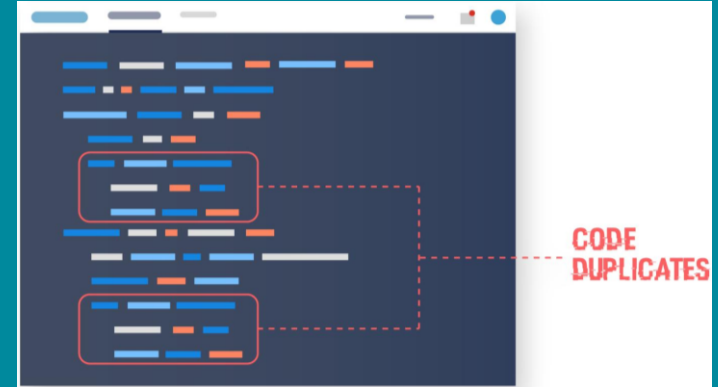
Even though we have used the waterfall method there still exists the possibility for the clients to have different requests at the end of the website design.



HANDOVER PROBLEMS

One handover problem we might face is that the evolution team might want to use an agile method. It could be more difficult for them because they might be unfamiliar with the plan-based method and they would have to start from scratch developing automated tests and the code might not be refactored or simplified as expected in the agile method.

Bad smells, coding errors or bad programming practices lead to code smells. These errors in the application code are frequently attributable to mistakes the application programmer made when writing it. Code smells are typically the result of standards not being followed when writing the code.



Some examples of bad smells are:

Duplicate code: where the same or similar code may be included in different places in a program.

Data clumping: This occurs when the same group of data items re-occurs in several places in a program.

Maintenance Cost Factors

Maintenance cost factors play a crucial role in the development of software. One significant factor is **team stability**, which has a direct impact on the overall quality of the software and helps to reduce errors. We as a software development team have maintained **consistency** from start to finish by getting a thorough understanding of the software's architecture, codebase, and functionality. The familiarity allows us to resolve any concerns quickly while maintaining a **high level of quality**.

Once the final product is delivered to the client, our team assumes contractual responsibility for its maintenance. This responsibility creates an incentive for us to design the software in a way that facilitates future changes. By considering the long-term maintenance aspect during the development phase, we can implement modular and scalable solutions. This approach makes it easier to **update, improve, and adapt** to changing user requirements. We can reduce future maintenance costs and ensure the software's adaptability and robustness by being proactive in our design choices.

Another vital aspect related to maintenance cost factors is the **skill set of the team members**. In our case, our team possesses the necessary skills and expertise required for efficient software maintenance. We have a diverse range of technical competencies, encompassing various programming languages, software development methodologies, and debugging techniques.





Thank You!



Do you have any
questions?