CARGO TRACING AND BUSINESS ANALYSIS

Group Id- 23

Guide- Prof. Kiran Kumari

1514068- Saurabh Baj
1514074- Niyati Daftary
1514080- Jayesh Gaur
1514085- Aditya Jawalikar

SOFTWARE DESIGN DOCUMENT

Contents

1 INTRODUCTION

1.1 Design Overview

This document presents the Software Design Document for the Cargo Tracing and Business Analysis project. This document gives insights about overview of the document. The requirement traceability matrix gives mapping about the requirements identified and the components in which they are fulfilled. With extensive research we chose Event driven architecture and Client Server Architecture as the architectures of our system. The system interface that is the hardware and software interfaces are mentioned in the document. Various components are described to a good extent which will help in making the system working. The document provides the description of user interface along with some of the images of the UI developed.

1.2 Requirement Traceability Matrix

Components / Requirements	Web Portal	RFID Scanner	Database
Trace Cargo	X	X	X
Business Analysis	X		X
Payment	X		
Login	X		X
Scanning RFID		X	X
Updation at Checkpoint			X
Register Seller	X		
Erasing RFID Data			X
View TimeLine	X		

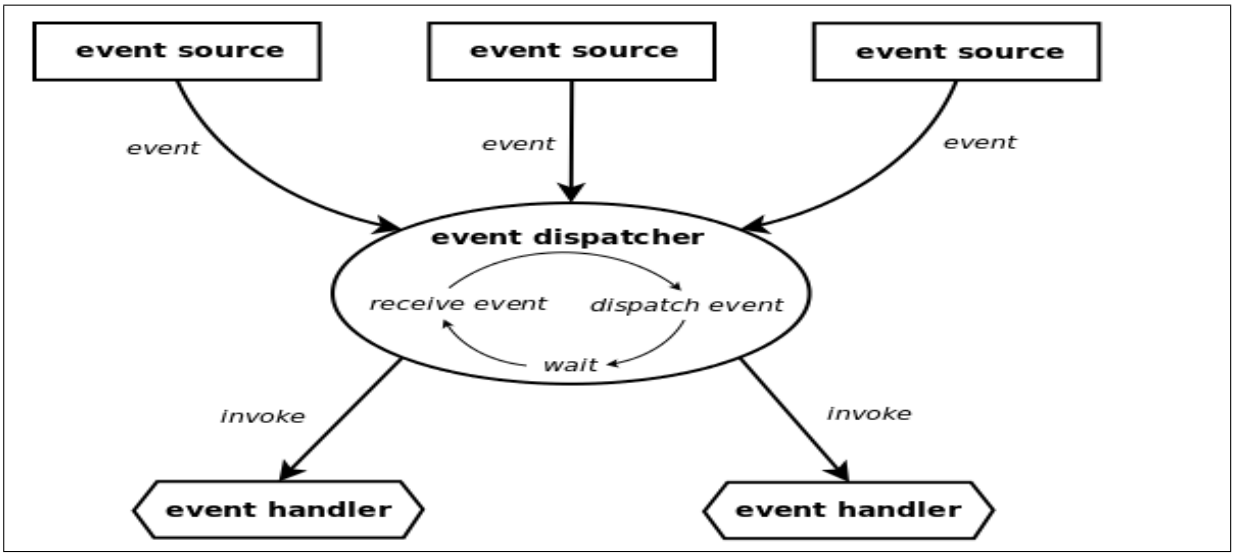
2 SYSTEM ARCHITECTURE

2.1 Chosen Architecture: Event Driven Architecture and Client Server Architecture

An event-driven application is a computer program that is written to respond to actions generated by the user or the system. In a computing context, an event is any identifiable occurrence that has significance for system hardware or software. As such, events include both user-generated actions like mouse clicks and keystrokes and system-generated events such as program loading or IDs passing through scanners.

Event-driven programming separates event-processing logic from the rest of a program's code.

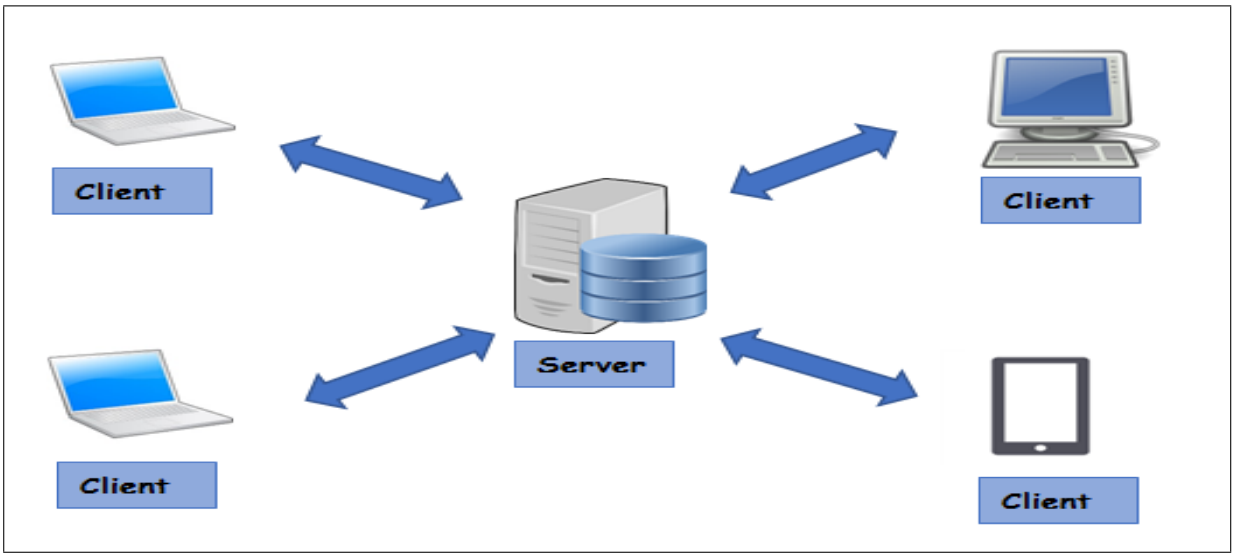
Figure 2.1: Event Driven Architecture



Client-server architecture, architecture of a computer network in which many clients (remote processors) request and receive service from a centralized server (host computer). Client computers provide an interface to allow a computer user to request services of the server and to display the results the server returns. Servers wait for requests to arrive from clients and then respond to them. Ideally, a server provides a standardized transparent interface to clients so that clients need not be aware of the specifics of the system (i.e., the hardware and software) that is providing the service. Clients are often situated at workstations or on personal computers, while servers are located elsewhere on the network, usually on more powerful machines.

This computing model is especially effective when clients and the server each have distinct tasks that they routinely perform. In our Cargo Tracing system, for example, a client computer can be running an application program for entering all the shipment related information while the server computer is running another program that manages the database in which the information is permanently stored. Many clients can access the server's information simultaneously, and, at the same time, a client computer can perform other tasks, such as sending e-mail. Because both client and server computers are considered intelligent devices, the client-server model is completely different from the old "mainframe" model, in which a centralized mainframe computer performed all the tasks for its associated "dumb" terminals.

Figure 2.2: Event Driven Architecture



2.2 System Interface Description

2.2.1 Hardware Interface

- The mandatory hardware requirement is a computer or an android device with basic android version of jellybean. This low specification is mainly due to the simplicity of design.
- Internet connection is a necessity.
- Decent RAM size and storage space required.
- RFID
- RFID Scanners

Figure 2.3: RC552 Scanner

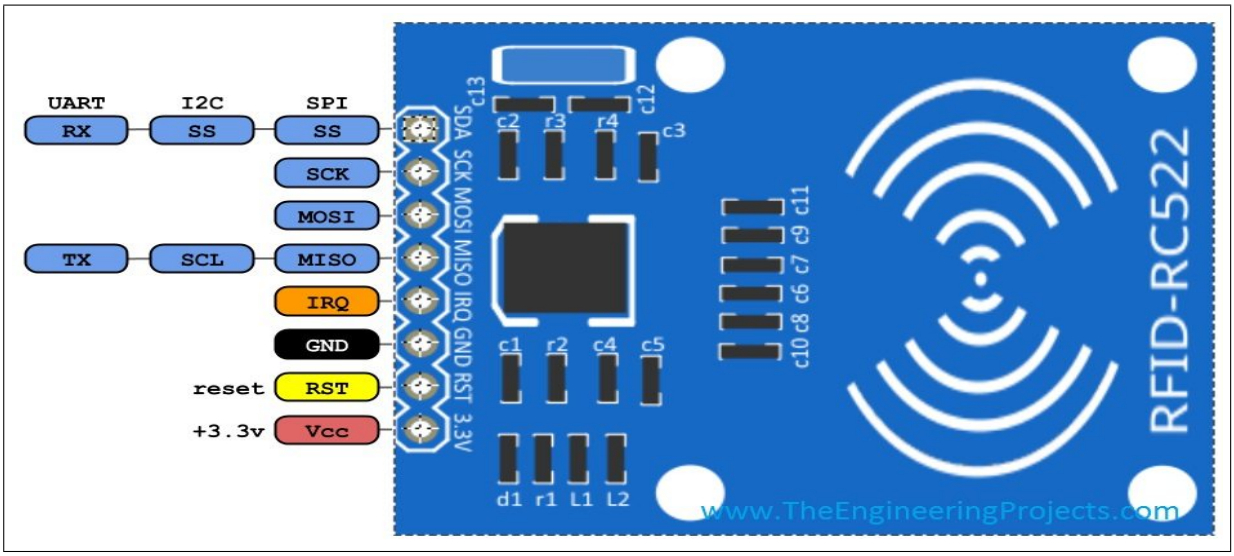
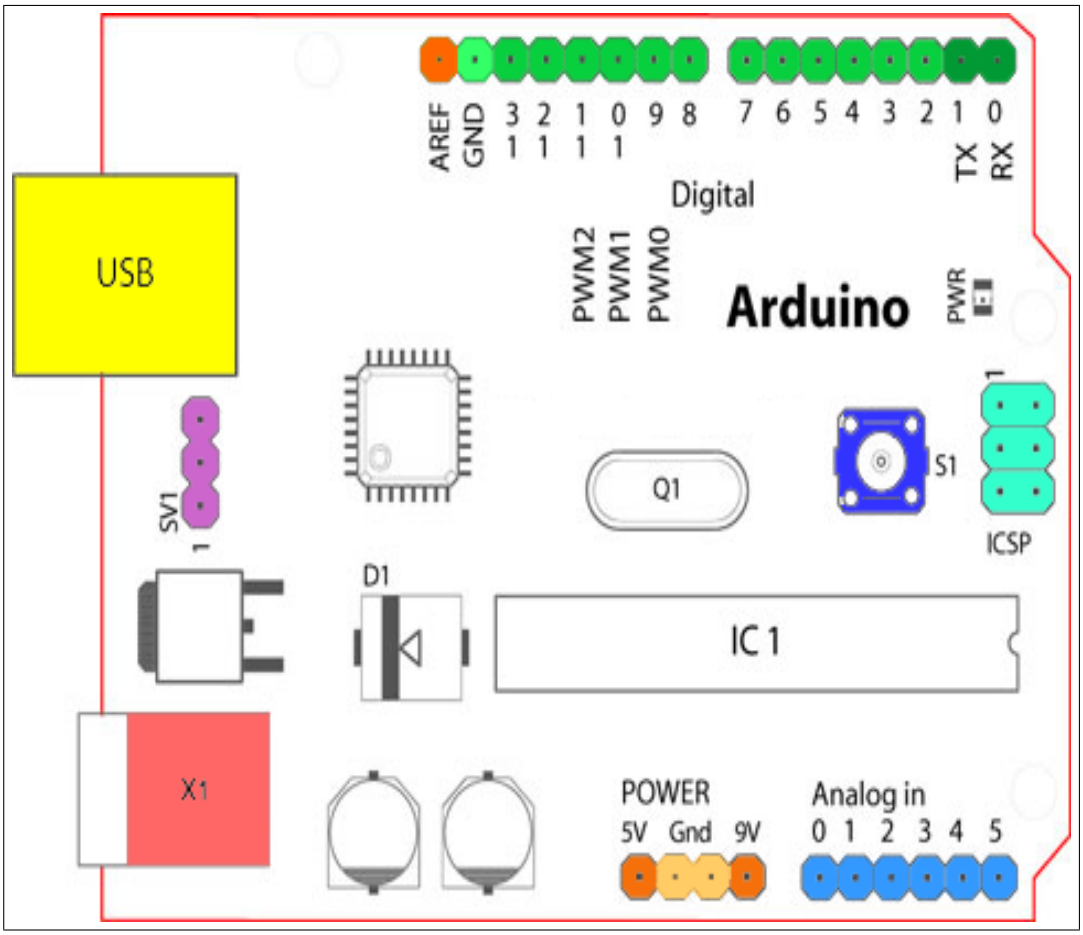


Figure 2.4: Arduino



- Aurdino Board

2.2.2 Software Interface

- A web browser with minimum compatibility of JavaScript.
- An android device with minimum android version of jellybean.

3 DETAILED DESCRIPTION OF COMPONENT

3.1 Server Connection

This component consists of php files which are downloaded locally on the hardware along with some on the server. These files help to connect with the sever and pass on the data on the server side from the client side in the form of files like text or xml. Thse files are interpreted by the server side and response in the form of JSON is given back to the client by the php files. These files also verify the data sent. Server side scripting and validation is also used.

3.2 Verification

Although there are php files to verify before sending the data, the software must contain verifying files for the data to be sent and received. Moreover the device must make sure that the data is sent by a human and not by any illegal practice. Basic validation also must be done separately of every data.

3.3 RFID tags

RFID tagging is an ID system that uses small radio frequency identification devices for identification and tracking purposes. An RFID tagging system includes the tag itself, a read/write device, and a host system application for data collection, processing, and transmission. An RFID tag (sometimes called an RFID transponder) consists of a chip, some memory, and an antenna.

RFID tags in the project will be used to track shipments, assigning them user data, unique identity to each shipment for scanning and tracking purposes.

3.4 RFID Reader

For interacting purpose of RFID tags, we have developed our own RFID scanning mechanism with hardware: RC522 - RFID Reader / Writer 13.56MHz includes a 13.56MHz RF reader cum writer module that uses an RC522 IC and two S50 RFID cards. The MF RC522 is a highly integrated transmission module for contact-less communication at 13.56 MHz. RC522 supports ISO 14443A/MIFARE mode. RC522 - RFID Reader features an outstanding modulation and demodulation algorithm to serve effortless RF communication at 13.56 MHz.

3.5 Wifi Module

For interaction purpose with our website we are using ESP MCU Node8266 module basically, NodeMCU is an open source IoT platform. It includes firmware which runs on the ESP8266 Wi-Fi SoC from Espressif Systems, and hardware which is based on the ESP-12 module

3.6 Aurdino

Aurdino is a basic micro-controller used to handle all the hardware components present in the system, main underlying code of basic functionality of the system like scanning of ID tags, writing of ID tags, communication with wifi module to send or receive data to/from a centralized database of a website. Aurdino is an open-source computer hardware and software company, project and user community that designs and manufactures single-board micro-controllers and micro-controller kits for building digital devices and interactive objects that can sense and control objects in the physical and digital world. The micro-controllers are typically programmed using a dialect of features from the programming languages C and C++.

4 USER INTERFACE

4.1 Description

The User will have an option initially to sign up or to log in in an existing account. Further once the user has logged in to his account they can track their shipments on the website. The user gets an option to know the possible prediction of transaction by providing basic details of the transaction. The user gets an option to manage the account where he can change his account type, edit his information, manage his security options and many more.

4.2 Images

Figure 4.1: Home Page

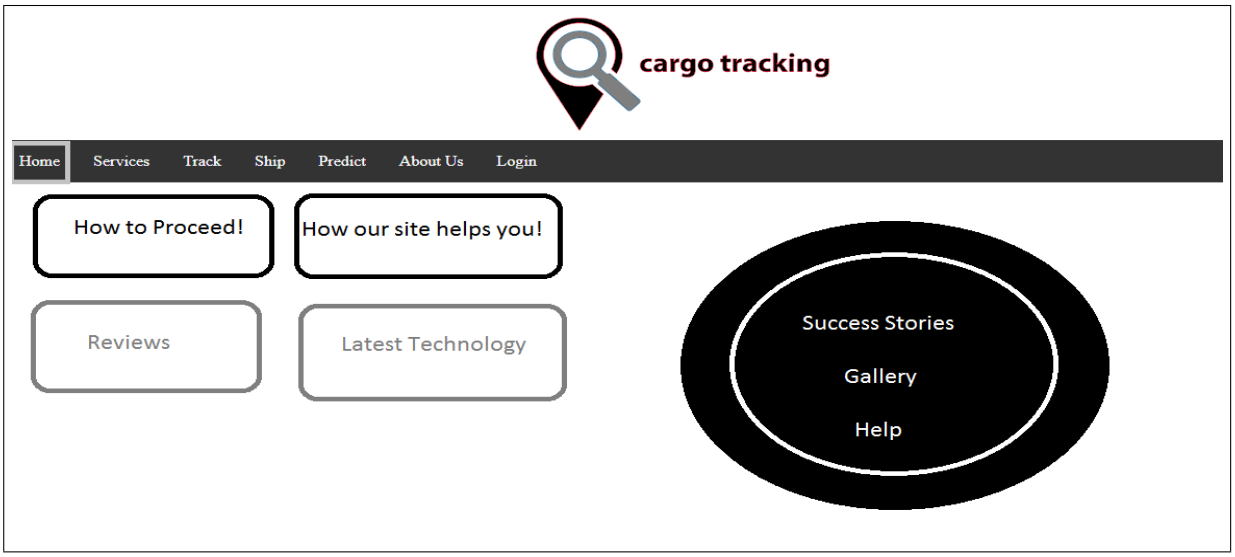


Figure 4.2: Tracking Page

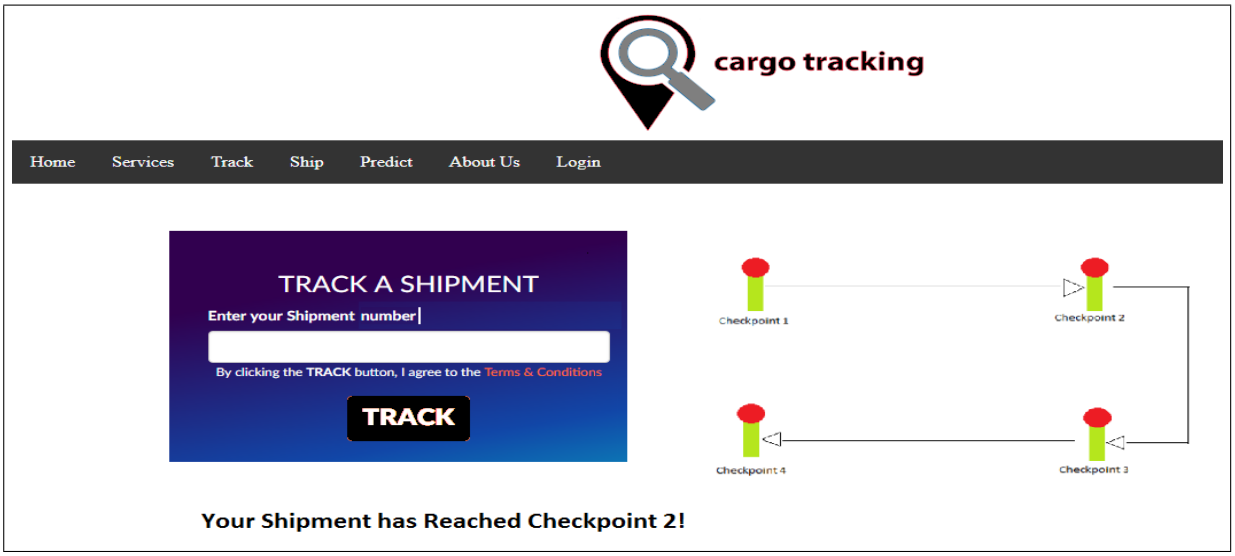


Figure 4.3: Business Analysis (Growth)