DEPARTMENT OF INFORMATION ENGINEERING
UNIVERSITY OF PADOVA

Web Applications

Nicola Boscolo
Mirco Cazzaro
Andrea Costa
Christian Marchiori
Marco Martinelli
Farzad Shami
Fabio Zanini

OUR GROUP

Our web application is designed to help companies and professionals manage their business activities efficiently while staying compliant with Italian regulations.

0110

bitsei

Business InTegrated System for Electronic Invoicing

MAIN FEATURES

INVOICING

Generate XML invoices that comply with Italian regulations

TRACKING

Keep track of customer activities and manage their data

INSIGHTS

Improve financial decision-making with informed insights

TABLE OF CONTENTS

1

ABOUT THE PROJECT

Inspiration and Purpose

1. INSPIRATION AND PURPOSE

Evolution of Electronic Invoicing in Italy

2015

Electronic Invoicing became a mandatory requirement for most of the companies

2022

Electronic Invoicing became mandatory also for sole proprietorships on a flat-rate basis

1. INSPIRATION AND PURPOSE

Evolution of Electronic Invoicing in Italy

2015

Electronic Invoicing became a Mandatory requirement for most of the companies

2022

Electronic Invoicing became Mandatory also for sole proprietorships on a flat-rate basis

Increased demand for reliable Electronic Invoicing softwares

1. INSPIRATION AND PURPOSE

Specialized Solution for Electronic Invoicing

Existing software solutions served multiple purposes

Lack of specialization in the market

Our platform combines:

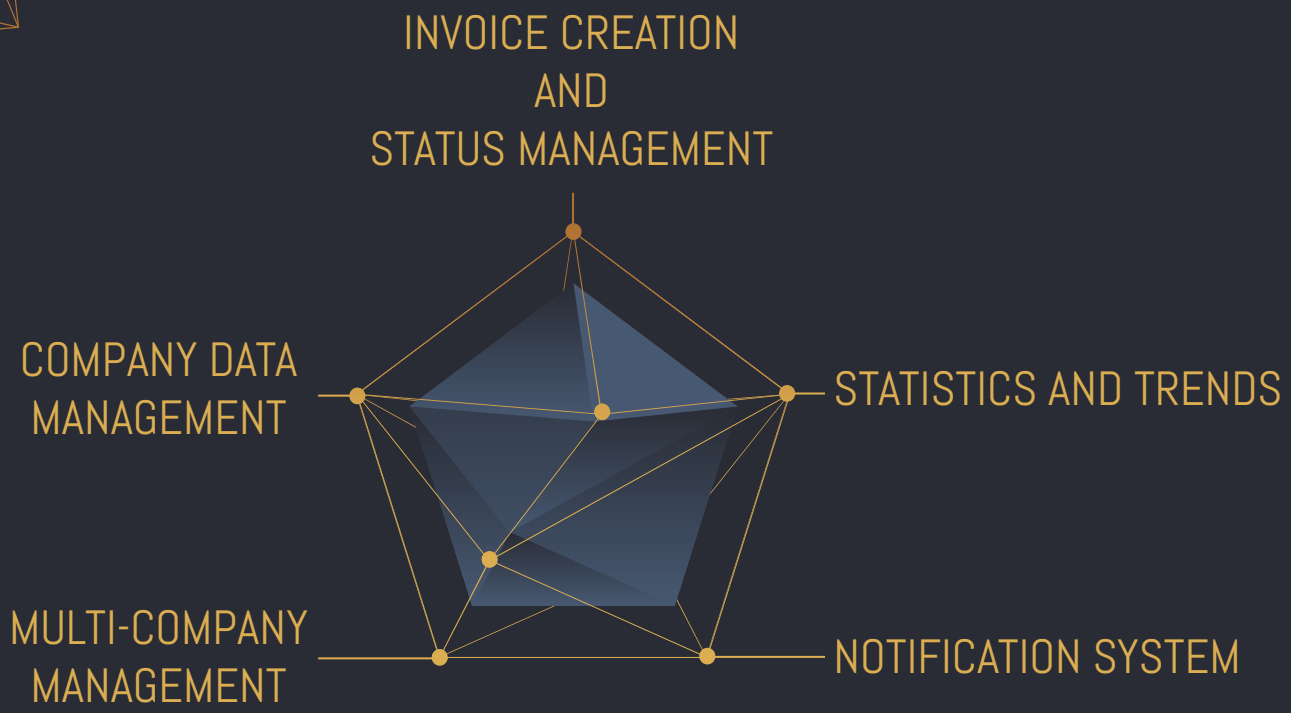
- Comprehensive Management
- Insightful Analysis
- Electronic Invoicing Functionalities

1

ABOUT THE PROJECT

Requirements Analysis

1. MAJOR REQUIREMENTS



1. INVOICE MANAGEMENT

STAGE 1
Create the invoice associated to a customer

STAGE 2
Add the products associated to the invoice

STAGE 3
Close the invoice and send the warning to the customer

STAGE 4
Wait for the customer to complete the payment

STAGE 5
Generate the invoice and send it to the customer

STAGE 6
Download the *XML* associated to the invoice

2

DATABASE

The design of our system

2. ENTITIES

OWNER

- **Owner ID**
- First Name
- Last Name
- Username
- Password
- Email
- Telegram ID

COMPANY

- **Company ID**
- ***Owner ID***
- Business Name
- Vat Number
- Tax Code
- Address Fields
- Unique Code
- Has Notifications

CUSTOMER

- **Customer ID**
- Business Name
- Vat Number
- Tax Code
- Address Fields
- Email
- PEC
- Unique Code

2. ENTITIES

INVOICE

- **Invoice ID**
- ***Customer ID***
- Status
- Warning Data
- Invoice Data
- Total
- Discount
- Pension Fund Refund
- Has Stamp

PRODUCT

- **Product ID**
- ***Company ID***
- Title
- Default Price
- Measurement Unit
- Description

INVOICE PRODUCT

- **Invoice ID**
- **Product ID**
- Quantity
- Unit Price
- Related Price
- Related Price Description

BANK ACCOUNT

- **Bank account ID**
- ***Company ID***
- IBAN
- Bank account friendly name

3

BACKEND

The functioning core of our system

3. PACKAGES

DAO
Accessing and manipulating data

FILTER
Let RRs access the data of the session (verify authentication)

RESOURCES
Represent the entities of the DB

REST
Used to perform operations on resources (*GET*, *POST*, *PUT*, *DELETE*)

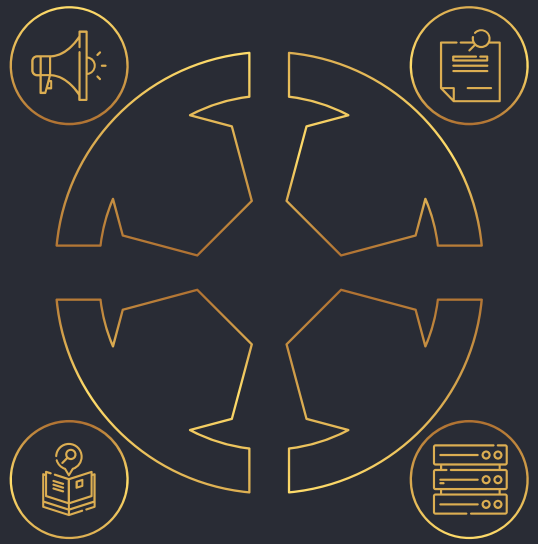
SERVLET
Also used to perform operations on resources

UTILS
Various util classes and functions

3. KEY PARTS

REST DISPATCHER SERVLET

Gets the request and calls the right rest resource



REST URI PARSER

Parses the *URI* to understand what's the request

REST RESOURCE

Forwards the request to the right *DAO*

DAO

Access the data and, if requested, modifies them

3. REQUESTS' WORKFLOW

STEP 1

A *Rest Request* is identified and sent to the *Dispatcher*

STEP 2

The *Dispatcher* forwards the *Request* to the appropriate *Rest Resource*

STEP 3

The *Rest Resource* in turn forwards the *Request* to the appropriate *DAO*

STEP 4

The *Rest Resource* receives the *DAO's Response* and write it in *JSON* format

STEP 5

The *Dispatcher* processes the *JSON* received by the *Rest Resource*

3. ACTIONS ON *"OWNER" (USER)*

CHANGE PASSWORD
Changes the login password for a user

RESET PASSWORD
Sends a token by mail to the user to reset his password

GET USER
Gets the attributes of an user

LIST USERS
Lists all the users registered in the system

LOGIN USER
Controls the authentication for a user

MANAGE NOTIFICATIONS
Manages email and telegram notifications

3. ACTIONS ON *"COMPANY"*

CREATE COMPANY
Creates a new company

DELETE COMPANY
Deletes an existing company

GET COMPANY
Gets the attributes of a company

GET COMPANY IMAGE
Retrieves the image (logo) of a company

LIST COMPANIES
Lists all the companies associated to the user

UPDATE COMPANY
Updates the attributes of a company

3. ACTIONS ON *"CUSTOMER"*

CREATE CUSTOMER
Creates a new customer

DELETE CUSTOMER
Deletes an existing customer

GET CUSTOMER
Gets the attributes of a customer

UPDATE CUSTOMER
Updates the attributes of a customer

LIST CUSTOMERS
Lists all the customers associated to a company

NOTIFY CUSTOMER
Notifies a customer via email or telegram

3. ACTIONS ON *"INVOICE"*

CREATE INVOICE
Creates a new invoice

DELETE INVOICE
Deletes an existing invoice

GET INVOICE
Gets the attributes of an invoice

UPDATE INVOICE
Updates the attributes of an invoice

LIST INVOICES (WITH FILTERS)
Lists all the invoices associated to a company, applying filters

GENERATE CHARTS
Generate charts relative to invoices data

3. ACTIONS ON *"PRODUCT"*

CREATE PRODUCT
Creates a new product

DELETE PRODUCT
Deletes an existing product

GET PRODUCT
Gets the attributes of a product

UPDATE PRODUCT
Updates the attributes of a product

3. ACTIONS ON *"PRODUCT"*

CREATE PRODUCT
Creates a new product

DELETE PRODUCT
Deletes an existing product

GET PRODUCT
Gets the attributes of a product

UPDATE PRODUCT
Updates the attributes of a product

3. ACTIONS ON *"INVOICE PRODUCT"*

⇧ SAME AS ABOVE

3. ACTIONS ON *"INVOICE - DOCUMENTATION"*

CLOSE INVOICE
Closes the invoice and generates warning pdf file

GENERATE INVOICE
Generates the xml and pdf files for the invoice

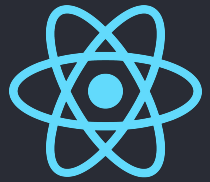
GENERATE CUSTOMERS REPORT
Generates a pdf customers list

GENERATE PRODUCTS REPORT
Generates a pdf products list

GET DOCUMENT
Gets a document (pdf or xml)

GET INVOICE DETAILS
Gets the details for an invoice

4



FRONTEND

The user-facing interface of our system

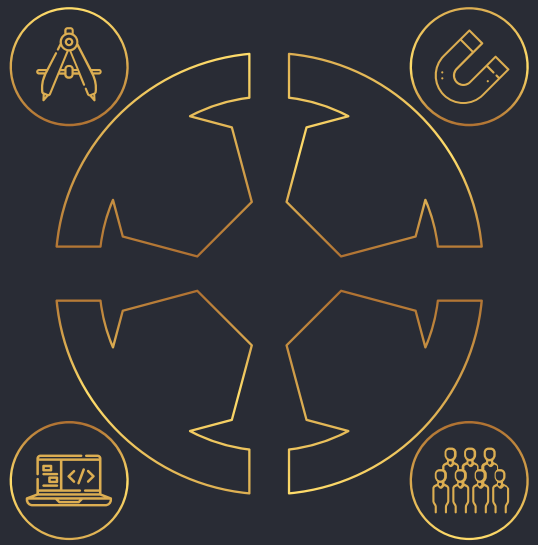
4. REACT: FRONTEND POWERHOUSE

COMPONENT-BASED ARCHITECTURE

React is built around a component-based architecture, which promotes reusability and modular development.

VIRTUAL DOM

React utilizes a virtual DOM that efficiently updates and renders only the necessary components when the underlying data changes.



DECLARATIVE SYNTAX

React uses a declarative syntax, allowing to describe how the user interface should look based on the current application state.

WIDELY USED

There are abundant resources, libraries, and tools available to support the development process.

4. MAIN LIBRARIES

REACT BOOTSTRAP

A widely-used library that combines the power of React with the styling and component library of Bootstrap.

AXIOS

A library that simplifies making HTTP requests

REACT HOOK FORM

A lightweight and flexible library for building forms in React.

REACT-ROUTER DOM

A widely-used library that provides declarative routing for React applications.

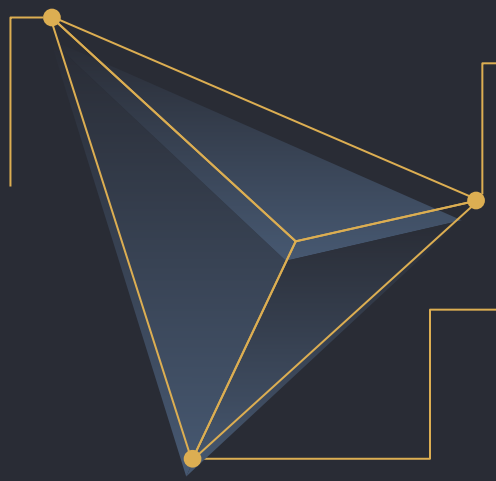
CHART JS

A JavaScript library that allows to create charts and graphs in web applications.

REDUX

A powerful state management library for JavaScript applications for managing application state.

4. MAIN LAYOUT



SIDEBAR

Provides a sidebar on the left with navigation options

NAVBAR

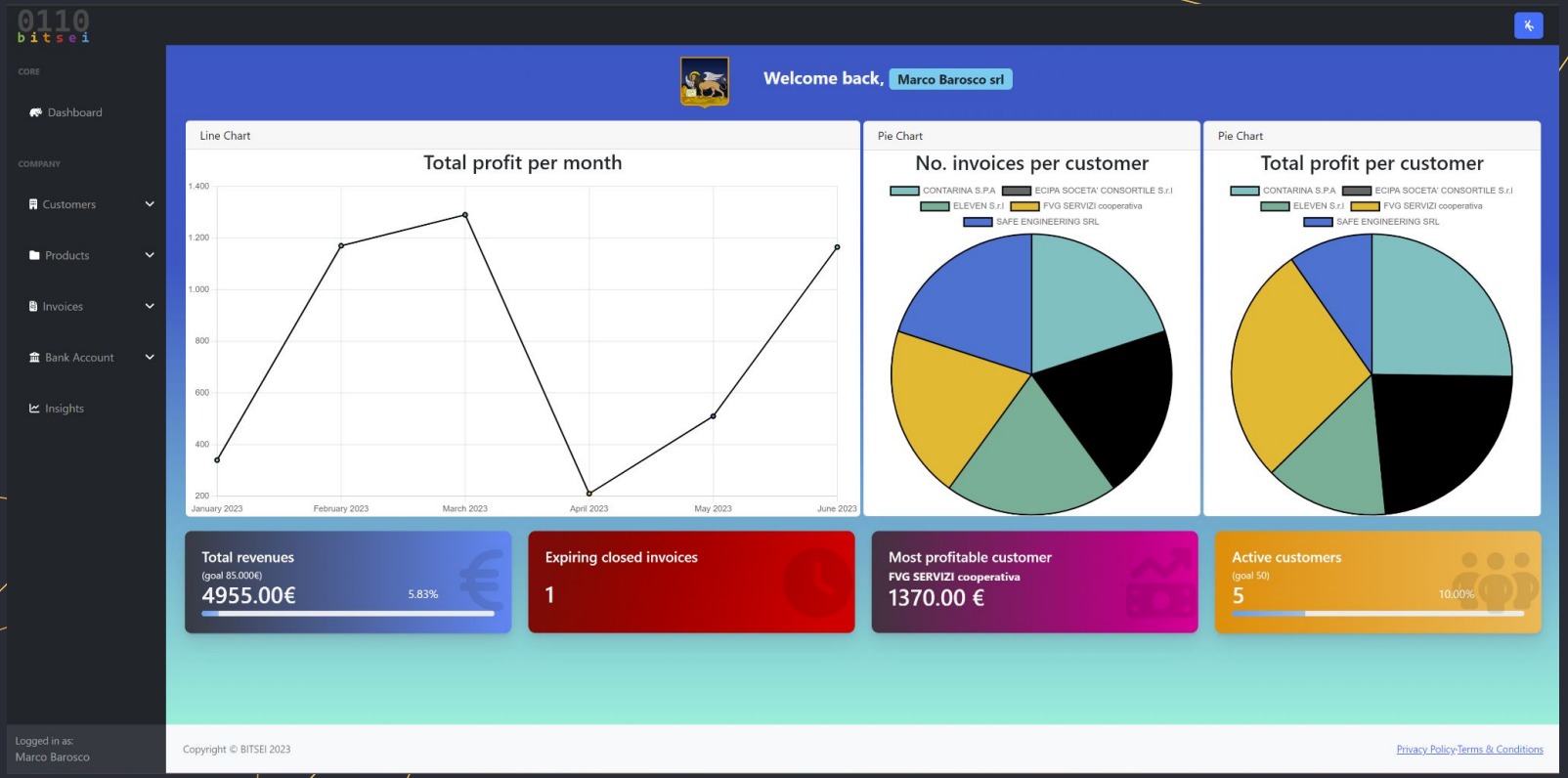
Contains Logo Image on the top left and Logout button on the top right

CONTENTS

The main content of the page. It may include Forms, Tables, and Charts

4. MAIN LAYOUT

Company Homepage



4. PAGES

LOGIN
Manages the login and authentication of an user

COMPANY
Manages the listing and editing of the user's companies

CUSTOMER
Manages the listing and editing of the company's customers

INVOICES
Manages the creation and handling of company's invoices

PRODUCTS
Manages the listing and editing of the company's products

BANK ACCOUNT
Manages the listing and editing of the company's bank accounts

INSIGHTS
Manages the creation of plots and charts using invoices' data

4. REQUESTS HANDLING

STEP 1

The *useState* hook is used to initialize a state variable data as an empty array.

STEP 2

UseEffect run after the initial render and every time the component re-renders. (Default Mode)

STEP 3

In the *UseEffect* it is defined the function to be called by the *Gate*

STEP 4

The *Gate* forwards the request to the Backend using *Axios* client

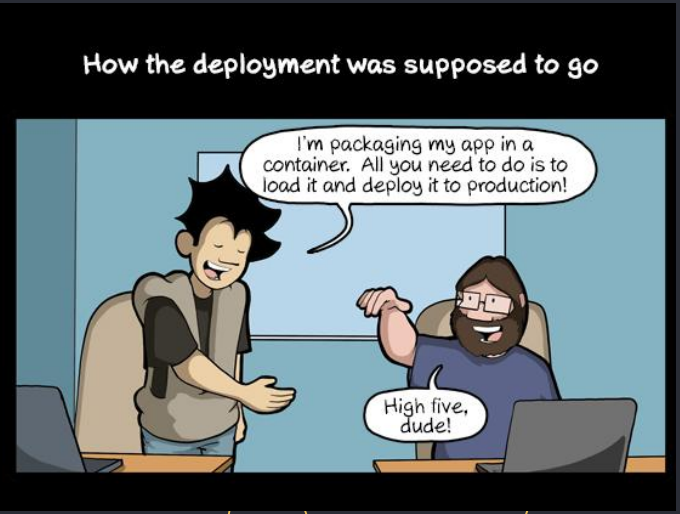
STEP 5

Inside the *UseEffect* it is received the response from the *Gate*

STEP 6

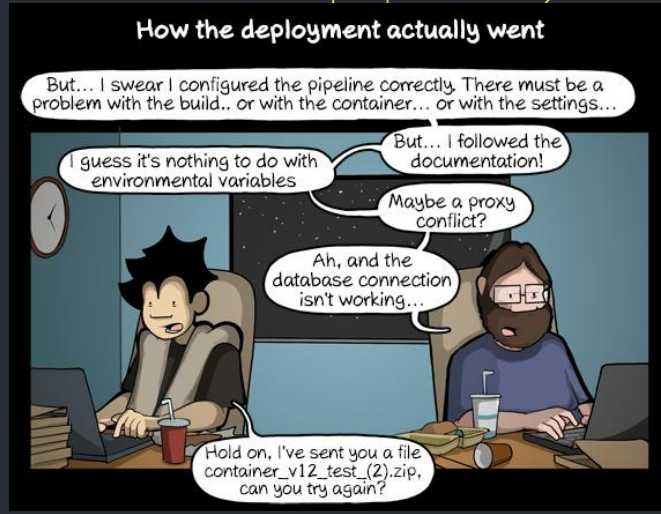
If the response has *status code == 200* the data are stored, in other cases an error is thrown 😔

4. DEPLOYMENT



Efficiency and Productivity

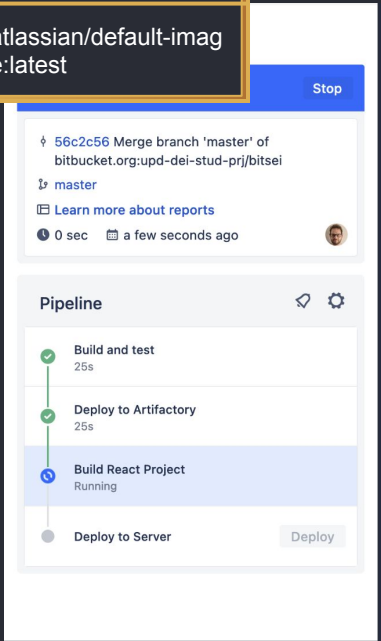
Error Handling and Monitoring

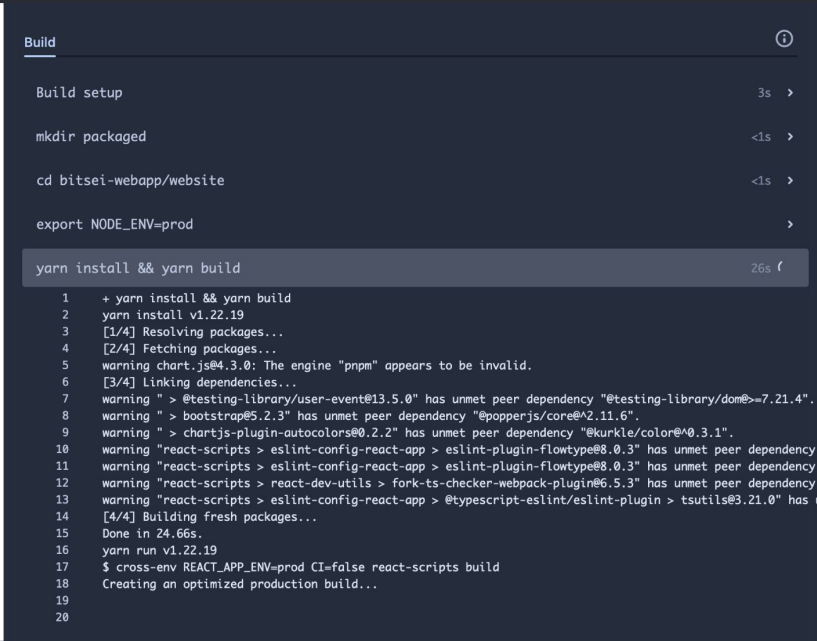


4. BITBUCKET PIPELINES

BRANCHES	STEP NAME	IMAGE
Master	Build and test	Maven 3.9.0
	Deploy to Artifactory	Maven 3.9.0
Client-side	Build React Project	Node 18.16.0
	Deploy to Server	atlassian/default-image:latest

Bitbucket Pipelines





5

LIVE DEMO

Real-Time Demonstration of our Key Features

DEMO

Server.IT
PREMIUM HOSTING

Our Web Application is hosted in a private server.

You can find it at *bitsei.it*

DEPARTMENT OF INFORMATION ENGINEERING

UNIVERSITY OF PADOVA

THANKS!

DO YOU HAVE ANY QUESTION?