Software Design Document

“Distributed computer system for the management of an owner’s association”

Name (s): Costache Theodor Alexandru
Dragoi Edith Cristiana
Enoiu Andra Maria
Lab Section:
Workstation:

Date: (03/30/2021)

TABLE OF CONTENTS

1. INTRODUCTION

1.1. Purpose

The purpose of this document is to give a detailed description of the requirements for the project " Distributed Computer systems for the management of an owner's association". It will illustrate the purpose and complete declaration for the development of the system.
This document is written for a more general audience, this document is intended for individuals directly involved in the development of the application. This includes software developers, testers, project managers, teachers, users and team members.

1.2. Scope

This software will represent a web application representation of an owner's association. The purpose is to simplify the activity of managing the tenants and to ease the owner's access to information concerning their payments/invoices or anything related to the association.

1.3. Overview

This document provides information about the software system of the Tenants Association, describes some features about the designing implementation and its overall architecture.

The first part of the document is a short description of the application, some references, acronyms and definitions.

The second part represents a short overview of the application.

The third part of the document are some diagrams about the functionality of the program.

The fourth part is not applicable yet, as well as the fifth part.

The sixth part of the document presents an overview of the graphical interface and all of the functionalities of the website presented through mockups.

The seventh and the eighth parts are not applicable yet.

1.4. Reference Material

IEEE. IEEE Std 830-IEEE Recommended Practice for Software Requirements.

C# Net Application tutorial:

https://www.youtube.com/watch?v=GcFJjpMFJvI&ab_channel=TraversyMedia

C# tutorial:

https://www.youtube.com/watch?v=GhQdlIFylQ8&ab_channel=freeCodeCamp.org

Mysql database tutorial:

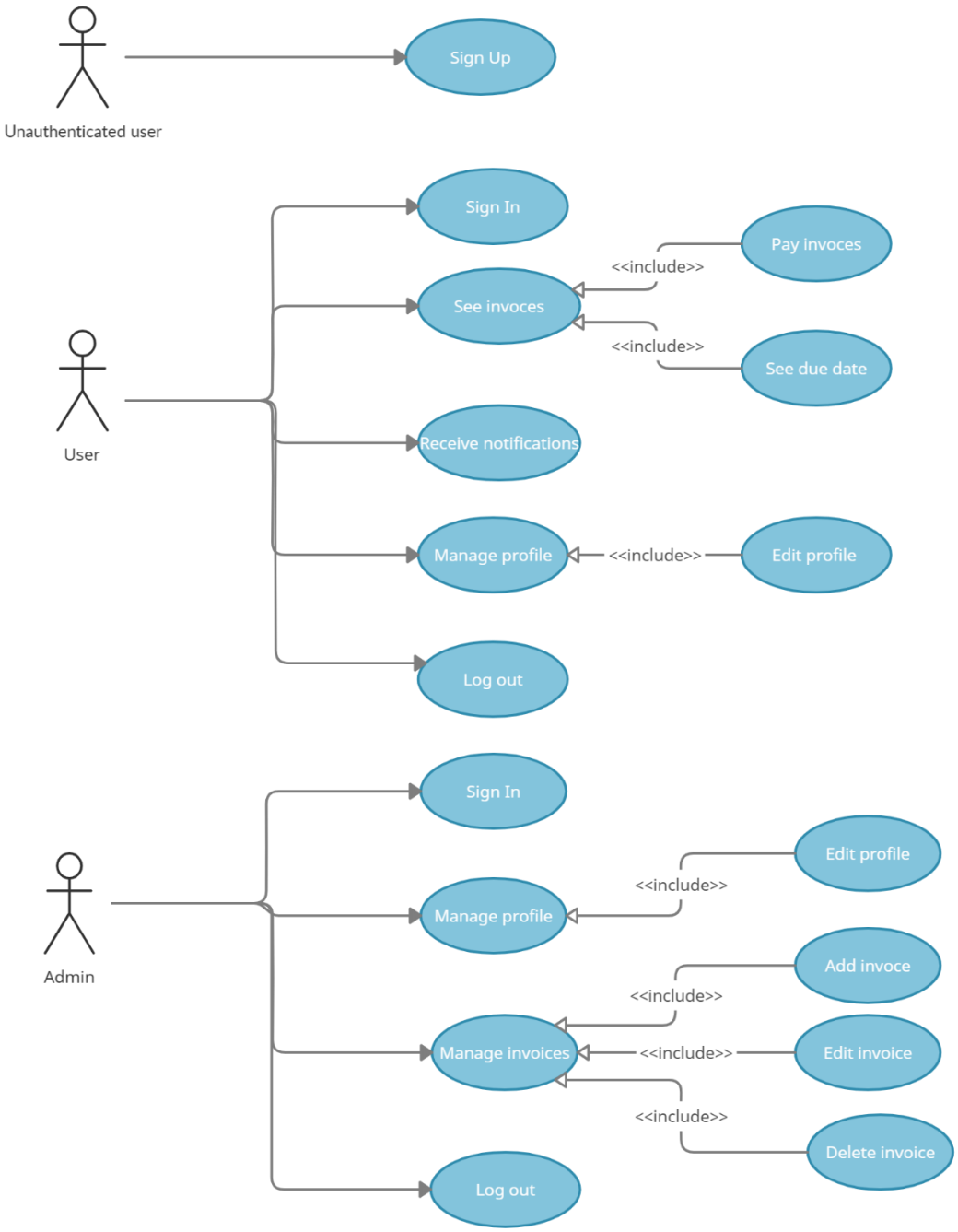
https://www.youtube.com/watch?v=7S_tz1z_5bA&ab_channel=ProgrammingwithMosh

1.5. Definitions and Acronyms

Abbreviation	**Meaning**
GUI	Graphical User Interface
SDD	Software Design Document
APP	Application
SQL	Structured Query Language
Server	a computer or computer program which manages access to a centralized resource or service in a network.
System	a group of devices or artificial objects or an organization forming a network especially for distributing something or serving a common purpose

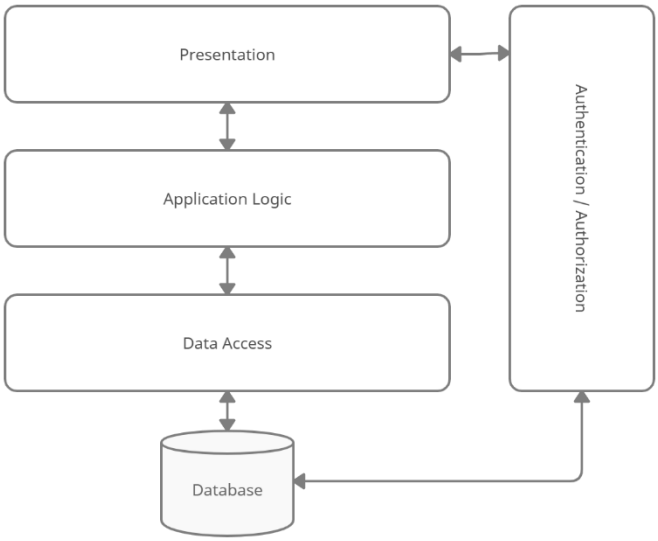
2. SYSTEM OVERVIEW

The tenants application is new, self-contained system intended for web application. This system is intended for clients who want to rent an apartment in a building, just with the help of the internet, from anywhere to everywhere. The users can rent a flat, see invoices and receive notification about their flat.



3. SYSTEM ARCHITECTURE

3.1. Architectural Design



The most common architecture pattern is the layered architecture pattern, otherwise known as the n-tier architecture pattern. This pattern is the de facto standard for most Java EE applications and therefore is widely known by most architects, designers, and developers.

Each layer of the layered architecture pattern has a specific role and responsibility within the application. For example, a presentation layer would be responsible for handling all user interface and browser communication logic, whereas a business layer would be responsible for executing specific business rules associated with the request. Also the presentation layer doesn't need to know or worry about how to get customer data; it only needs to display that information on a screen in particular.

The application layer doesn't need to be concerned about how to format customer data for display on a screen or even where the customer data is coming from; it only needs to get the data from the data access layer, perform application logic against the data (e.g., calculate values or aggregate data), and pass that information up to the presentation layer.

Data access layer in computer software is a layer of a computer program which provides simplified access to data stored in persistent storage of some kind, such as an entity-relational database.

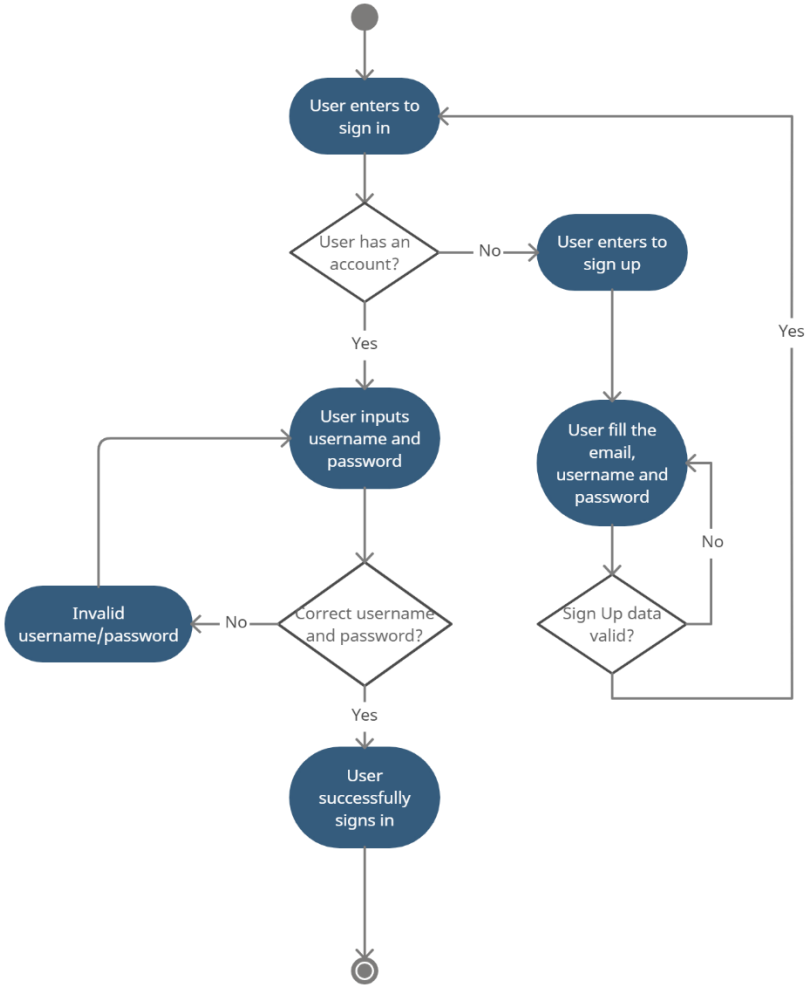
The database layer takes care of data access. An object from the database layer can write itself to one or more tables. In the database layer, you'll find things like database, connection, table, SQL, and result set.

Authentication is the process of verifying the identity of an individual. A unique identifier is associated with a user which is the username or userid. Traditionally, we use a combination of username and password to authenticate a user. A user can interact with a web application using multiple actions. Access to certain actions or pages can be restricted using user levels. Authorization is the process of controlling user access via assigned roles & privileges.

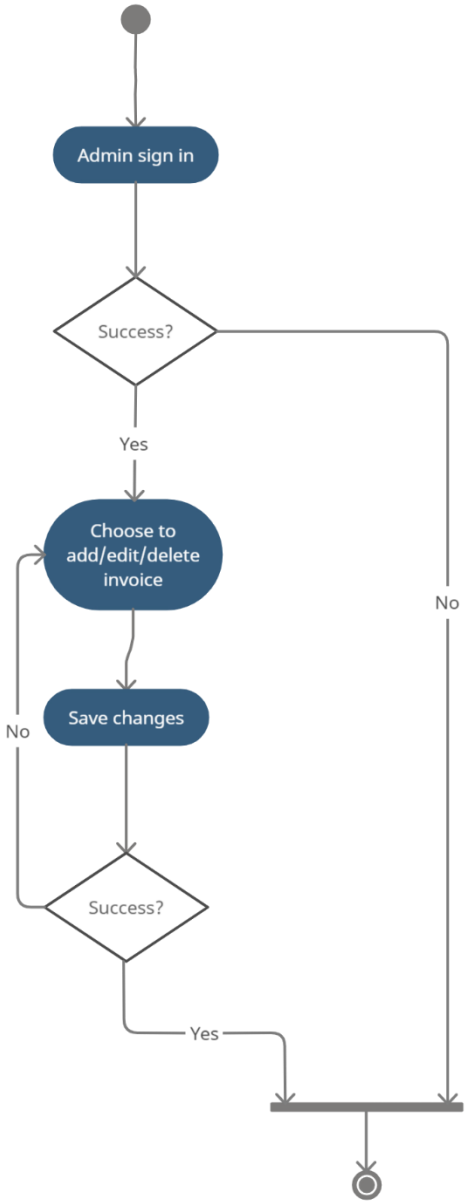
The layered architecture pattern is a solid general-purpose pattern, making it a good starting point for most applications, particularly when you are not sure what architecture pattern is best suited for your application.

3.2. Decomposition Description

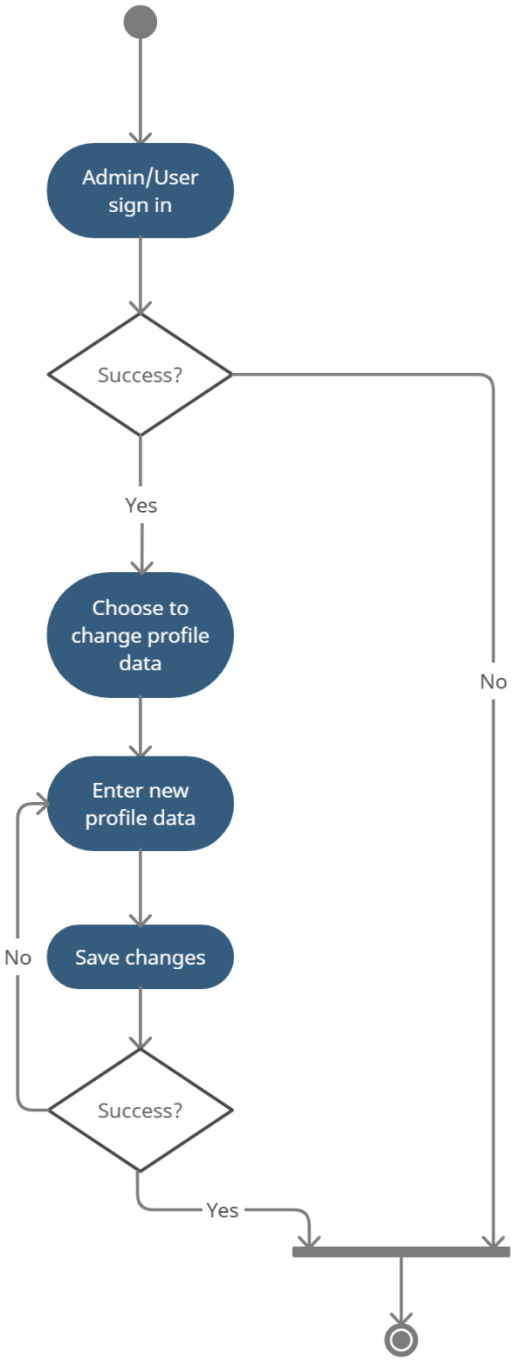
3.2.1. Sign up / Sign in



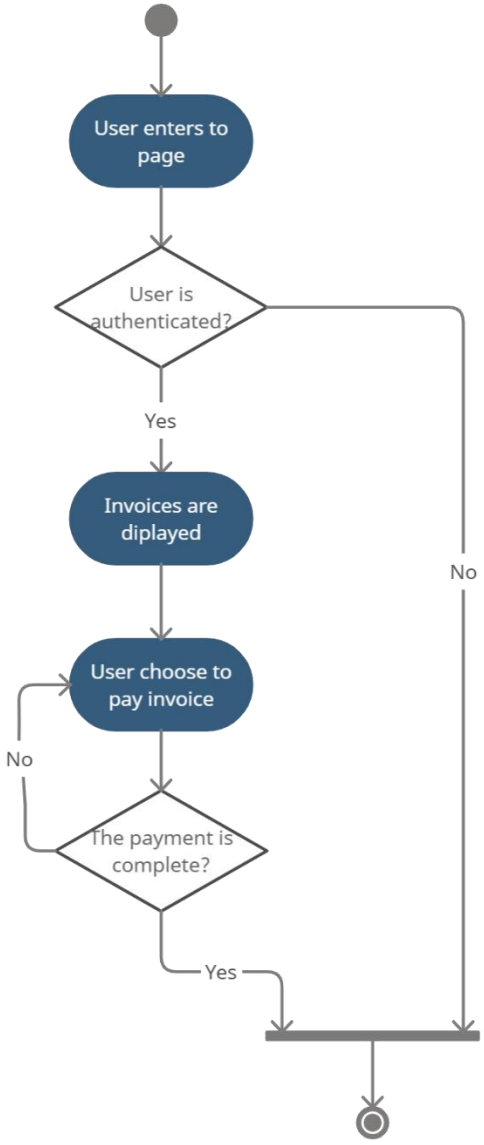
3.2.2. Admin manage invoices



3.2.3. Admin/User manage profile



3.2.4. User manage invoices

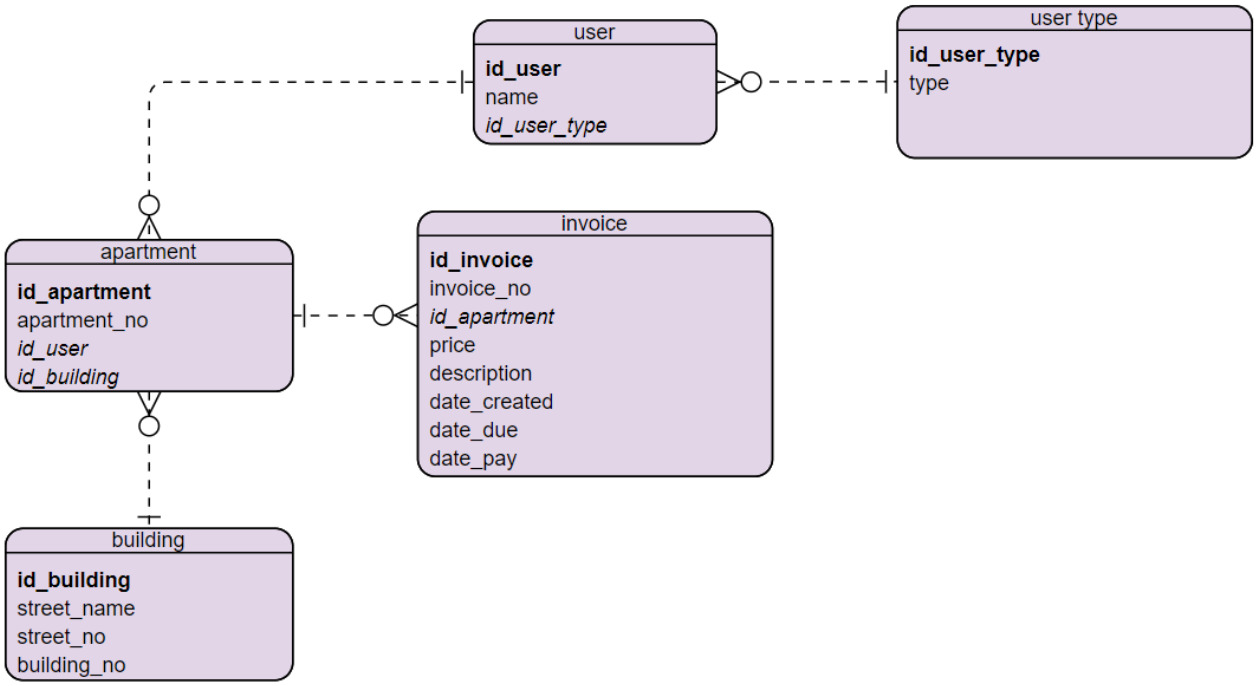


3.3. Design Rationale

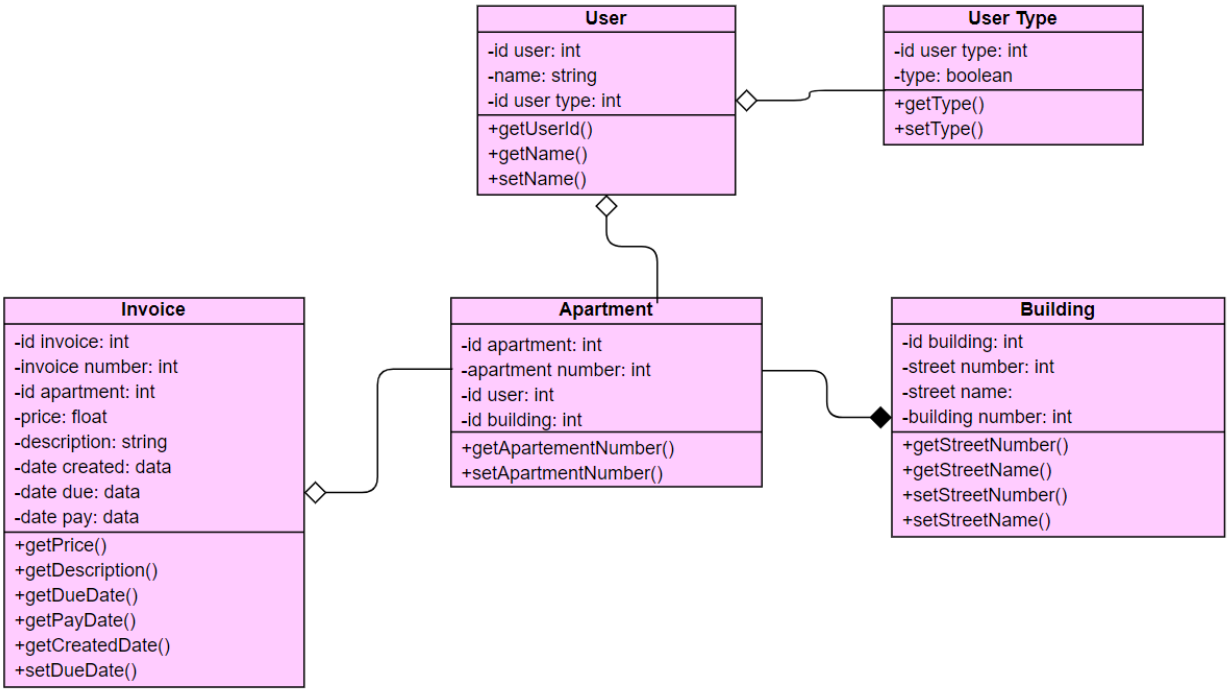
N/A.

4. DATA DESIGN

4.1. Data Description



4.2. Data Dictionary



5. HUMAN INTERFACE DESIGN

5.1. Overview of User Interface

There shall be 2 interfaces, one for the administrator and one for the user. A common interface, the login screen, which will have an input prompt for the username and password, which will be present for both roles.
The layout shall be as such:
The HEADER containing:

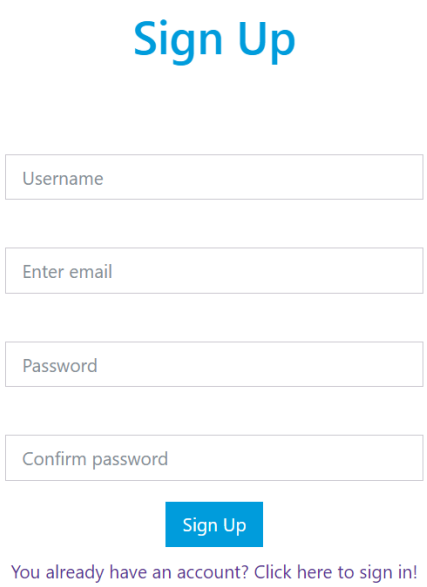
- The logo
- The currently logged in user's username.
- Buttons for log-out and account information ("My account").
- Button for the checking of invoices ("Invoices").
- Button for checking info pertaining to income/debts/expenditure (Services).

The body shall be different pertaining to the Invoices and Services buttons which shall take the user to a different UI depending on their role.
The body shall contain:

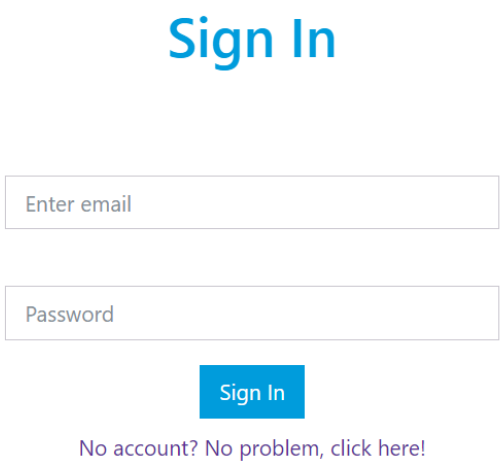
- The menu.
- The content.

5.2. Screen Images

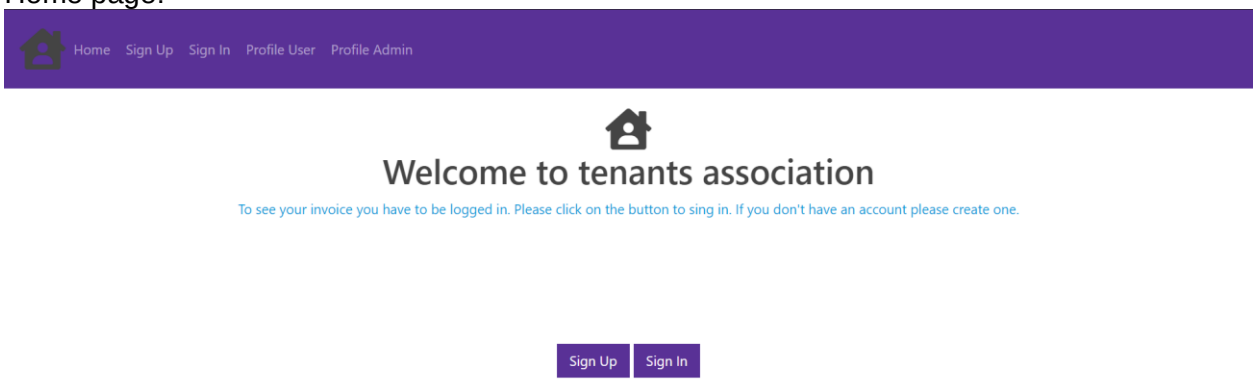
SignUp menu:



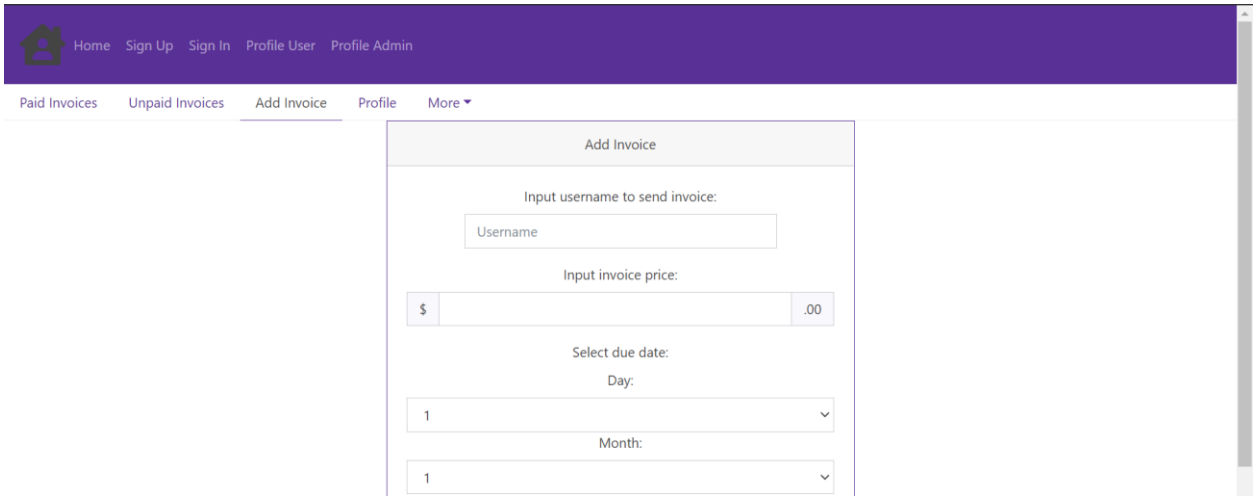
SignIn menu:



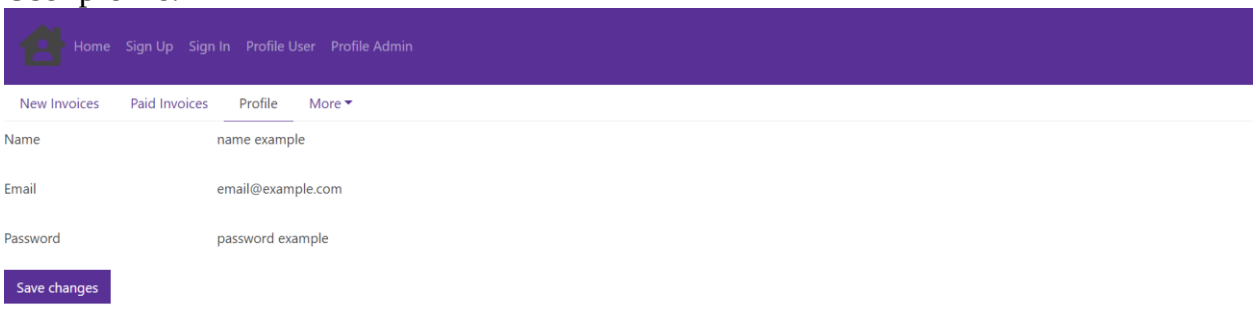
Home page:



Admin add invoice:



User profile:



6. REQUIREMENTS MATRIX

NA.

7. APPENDICES

NA.