ALKAYAN MUSIC WEB APP

CSE233: AGILE SOFTWARE ENGINEERING

Sohayla Ihab	19P7343	Development/PM
Mark Sameh	19P1863	Documentation
Salma Ihab	19P8794	Database
Mohammed Hekal	17P8071	JIRA/Trello
Noorhan Hatem	19P5821	Development/DB

CUSTOMER MEETING REPORT

The Objective

To develop a website for Kayan stakeholders.
AlKayan is a student activity run by Ain Shams University and sponsored by the university president Mahmoud Elmetainy, the vice president of student affairs Abdelfatah Seoud, and the Faculty of Specific Education dean Osama Elsayed. It was created to uncover and develop the musical talents of Ain Shams students. The first course, during August 2022, was a remarkable success, garnering over 300 participants from all across the city. The second course is due to start at 18th November, 2022. The staff supervising AlKayan will be our clients.

HIGH LEVEL REQUIREMENTS

Functions of the website include but are not limited to:

1) Registration
2) Payment
3) Mailing List
4) News/ Images

Application Form

All applicants must submit their information and agree to the terms and conditions. Info app is required to collect:

- **Personal Info Section:** Full name, Date of Birth, National ID and/or Passport
- **Education Info Section:** College/ University/ Institute, Current Year, Faculty/ Specialization
- **Contact Info Section:** Phone number, Email
- **Extra Info Section:**
  - Person with Disability: Name of guardian/caretaker, contact no., relation with person with disability
  - Extra info for staff to know (eg. allergies etc)
- **Instrument Section:** here, app must provide visually-aided (pictures, etc) detailed information of instruments available (guitar, piano, violin, oud, qanon, tabla, drums, flute, recorder, accordion, mandolin, xylophone, vocals western, childrens' choir) musical institute (TCL and ASU,) and paying tiers (one instrument free for ASU students for 3-month course,

otherwise paying 750 LE per 3-month course or additional instrument). Applicant then chooses his first option of instrument, and any more.

Communication

Website language available in English only. *

- **With applicant: Anyone can optionally sign-up to a mailing list.** ‘Contact Us’ WhatsApp pop-up in website. *
- Applicant and supervisor: applicant information is sent to رعاية الشباب in case ASU student.

Payment

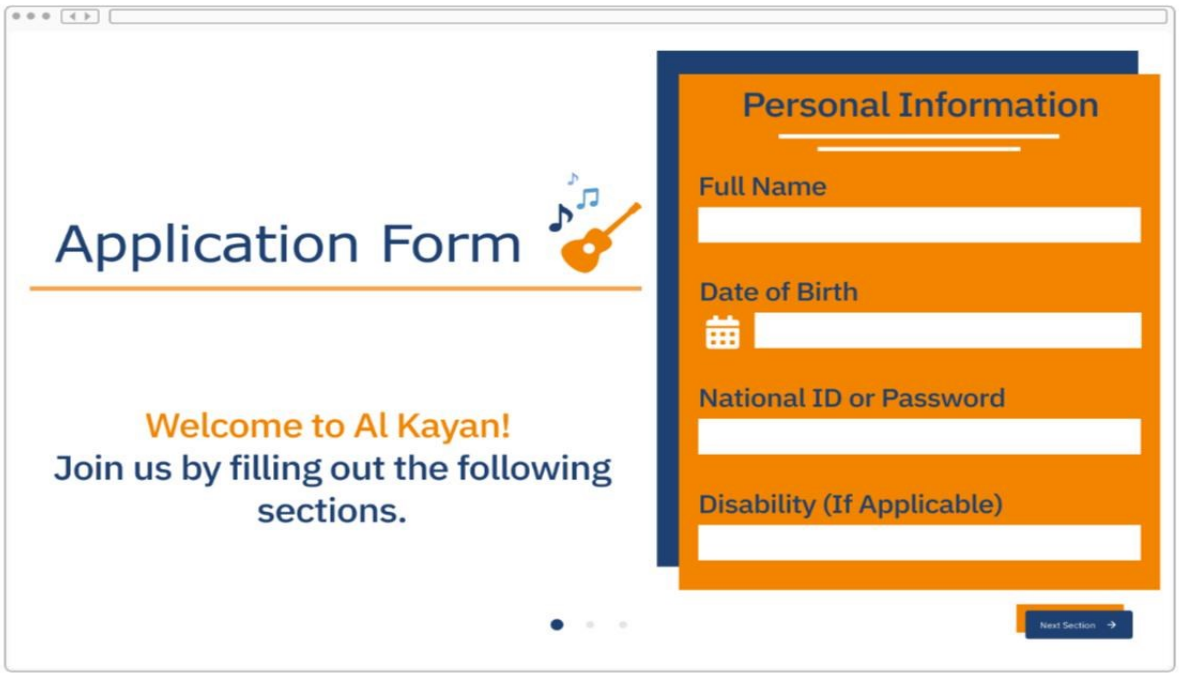
- **Ability to view and calculate different options + view paying types before paying. (transparency)**
- **Open to donations.**

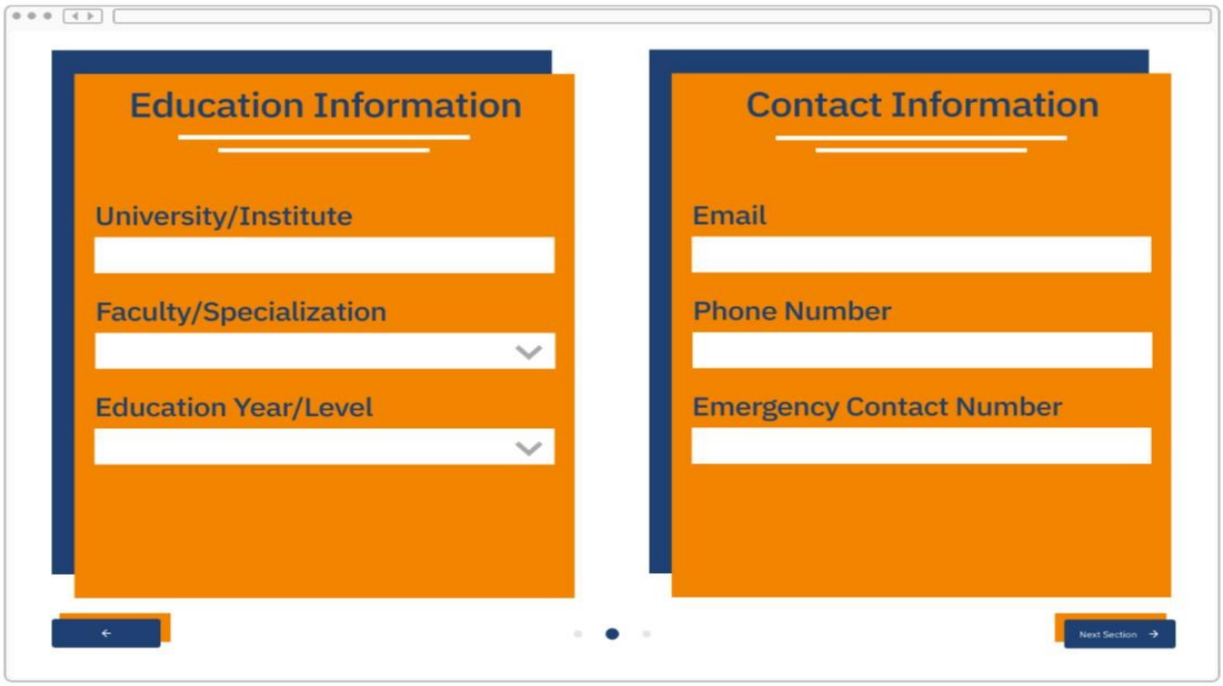
Other requirements

‘About Us’ section in website. * Supervisor info and social media and groups. * Blogs. * Events and seat booking. *

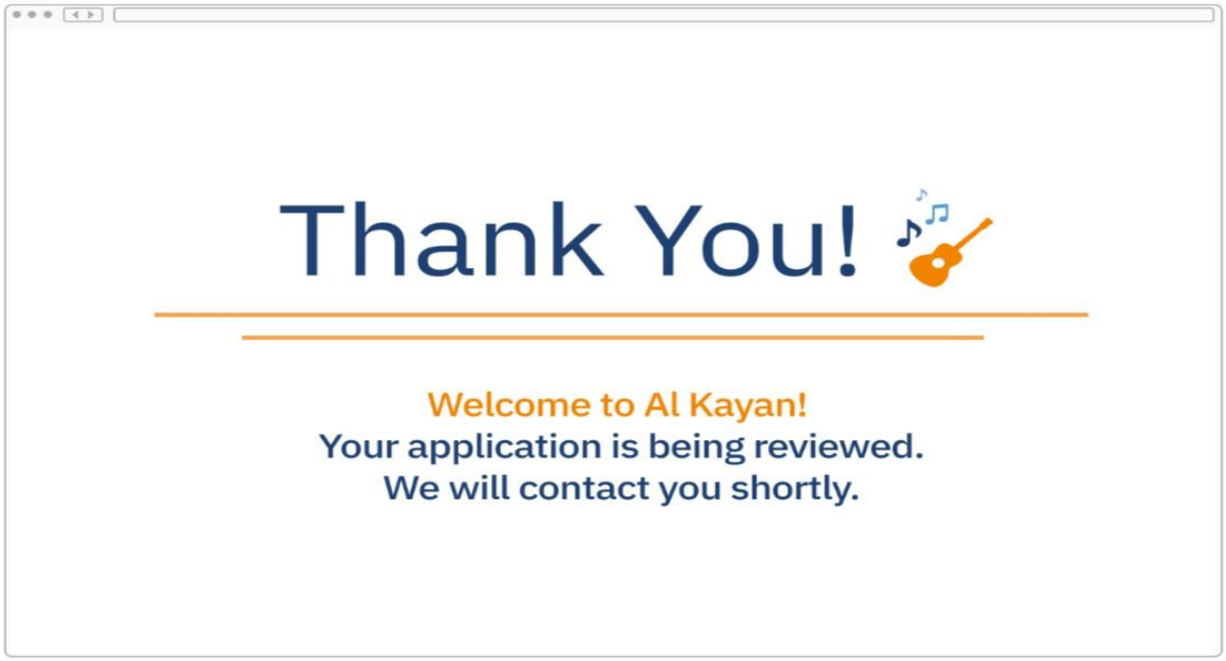
***Note: these requirements are non-functional.**

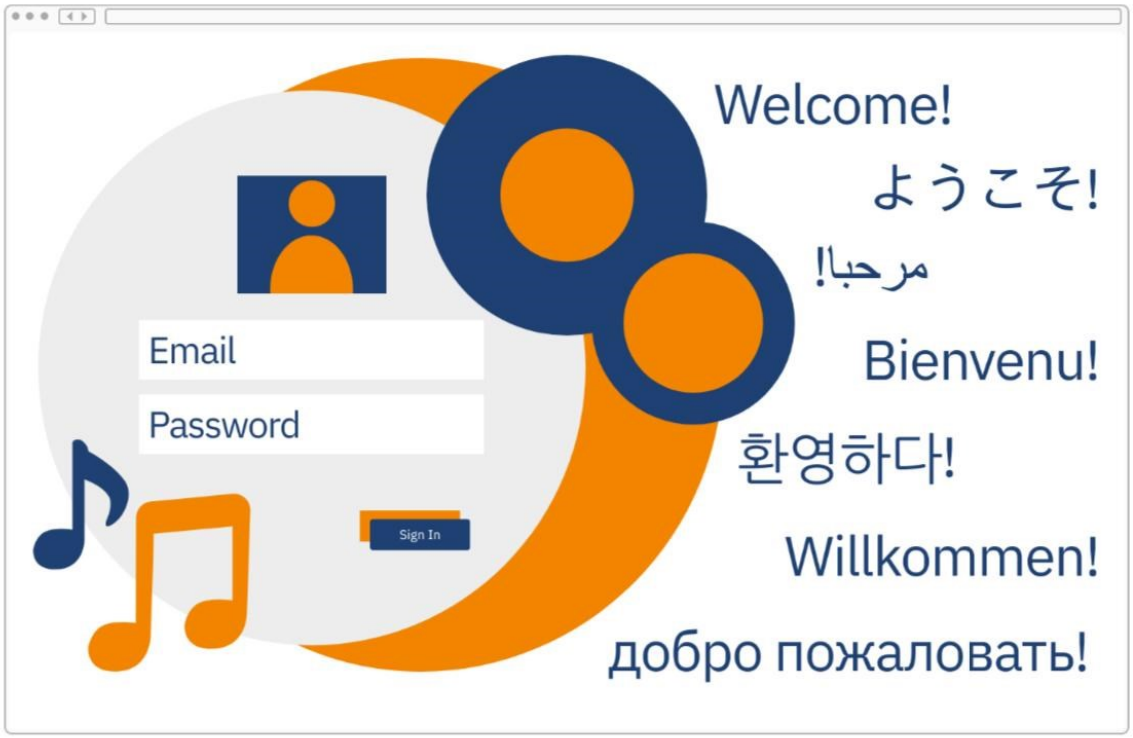
PROTOTYPE


Application Form
Welcome to Al Kayan!
Join us by filling out the following sections.
Personal Information
Full Name
Date of Birth
National ID or Password
Disability (If Applicable)
Next Section

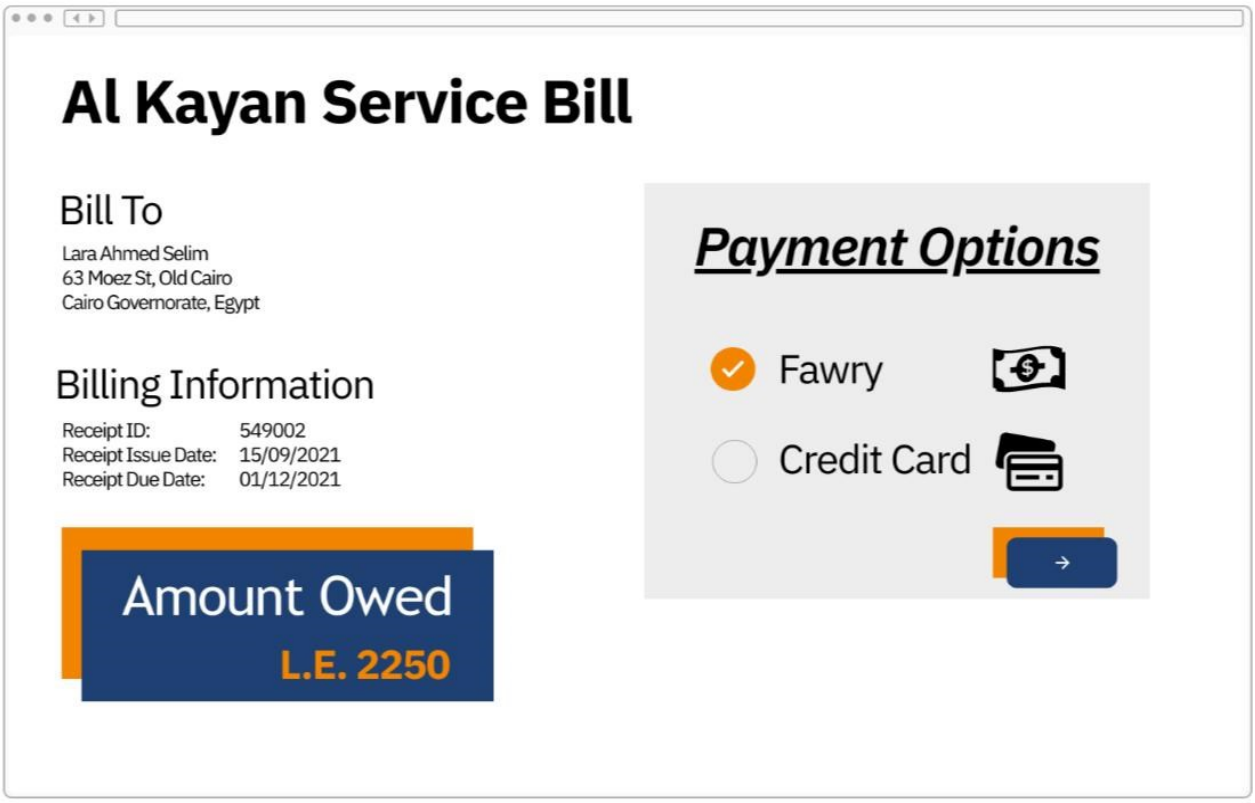

Education Information
University/Institute
Faculty/Specialization
Education Year/Level
Contact Information
Email
Phone Number
Emergency Contact Number
Next Section


Select Your Training(s)
Instruments
Guitar
Tabla
Drums
Xylophone
Piano
Qanun
Oud
Accordion
Violin
Mandolin
Flute
Recorder
Vocals
Western Vocals
Children's Choir
Musical Institute
Each course costs 750 LE and is 3-months long.
ASU students are awarded their first course for free!
Complete Registration

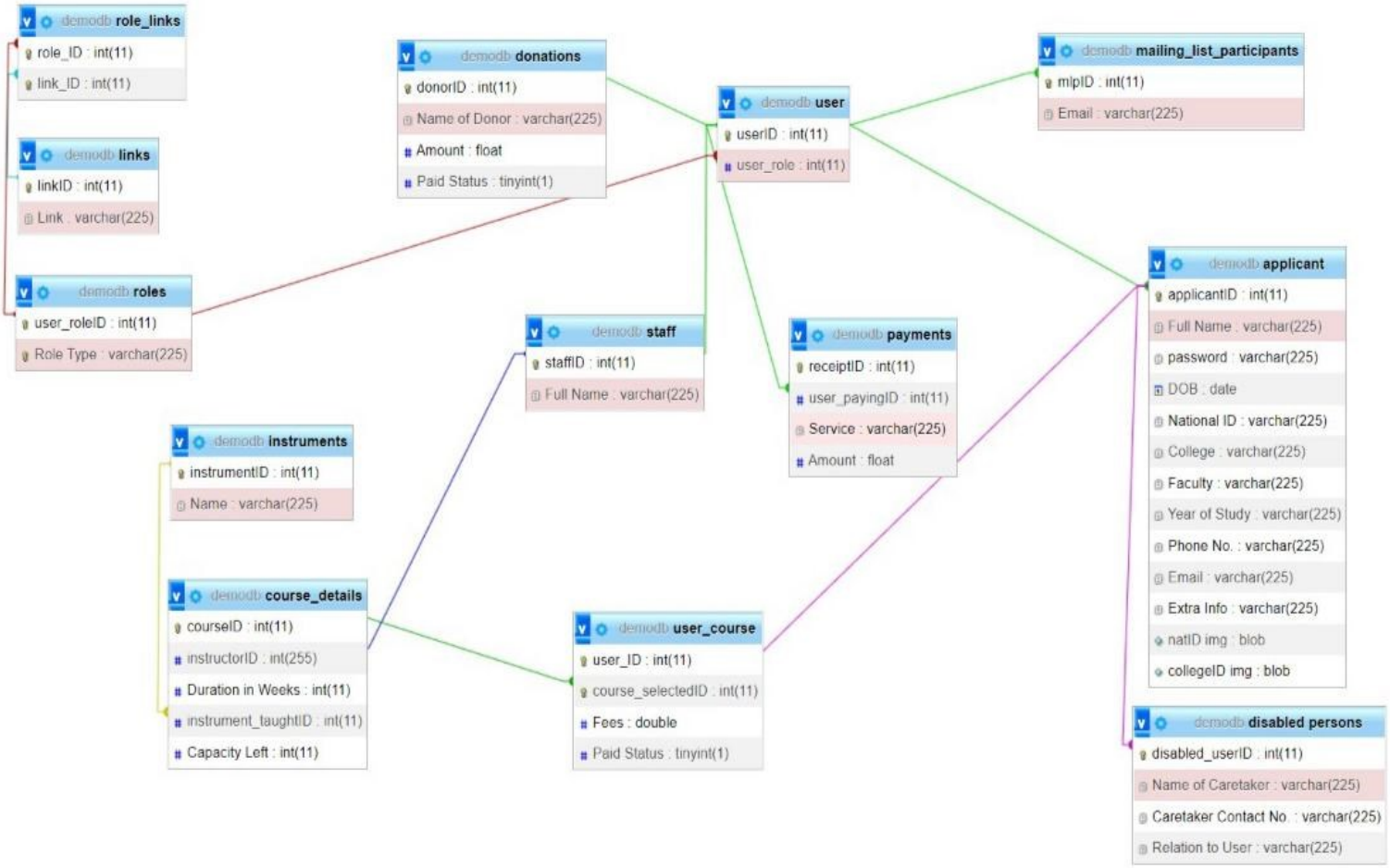

Thank You!
Welcome to Al Kayan!
Your application is being reviewed.
We will contact you shortly.


Email
Password
Sign In
Welcome!
ようこそ!
مرحبا!
Bienvenu!
환영하다!
Willkommen!
добро пожаловать!


Hello Lara!
How Can We Help You Today?
View Schedule
View Receipt
Cancel Registeration
Contact An Administrator
Register New Course


Al Kayan Service Bill
Bill To
Lara Ahmed Selim
63 Moez St, Old Cairo
Cairo Governorate, Egypt
Billing Information
Receipt ID: 549002
Receipt Issue Date: 15/09/2021
Receipt Due Date: 01/12/2021
Amount Owed
L.E. 2250
Payment Options
Fawry
Credit Card

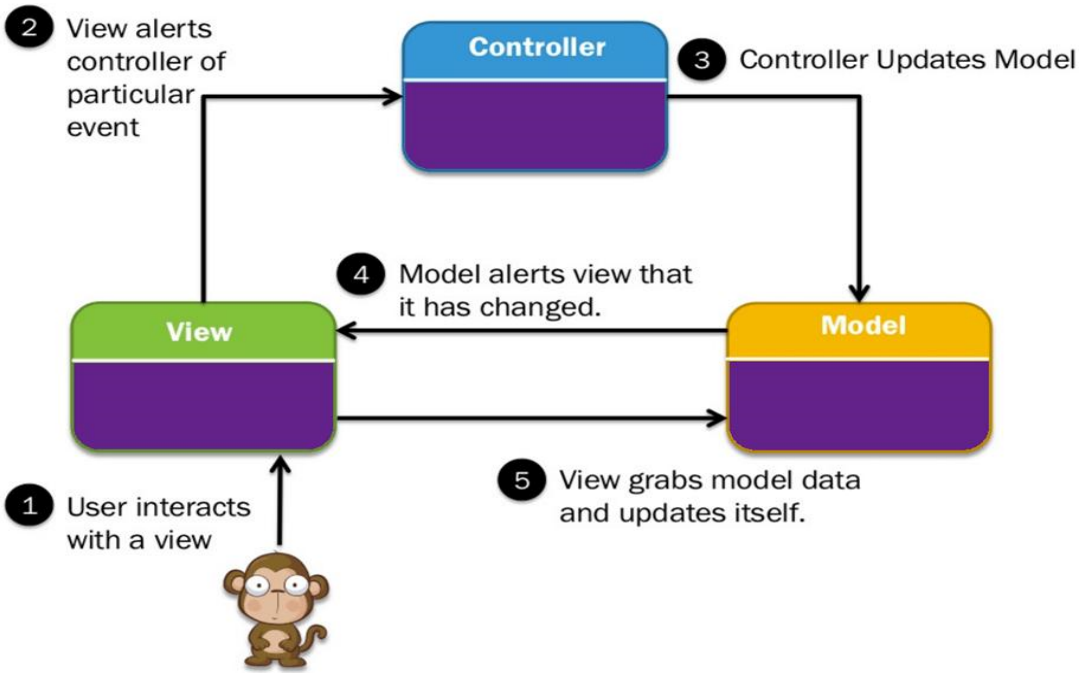
DB SCHEMA



USING MVC

The Model-View-Controller (MVC) framework is an architectural pattern that separates an application into three main logical components Model, View, and Controller. Each architecture component is built to handle specific development aspect of an application.
MVC separates the business logic and presentation layer from each other. It was traditionally used for desktop graphical user interfaces (GUIs). Nowadays, MVC architecture in web technology has become popular for designing web applications as well as mobile apps.

Here is the detailed architecture of MVC framework:



Model, View, Controller

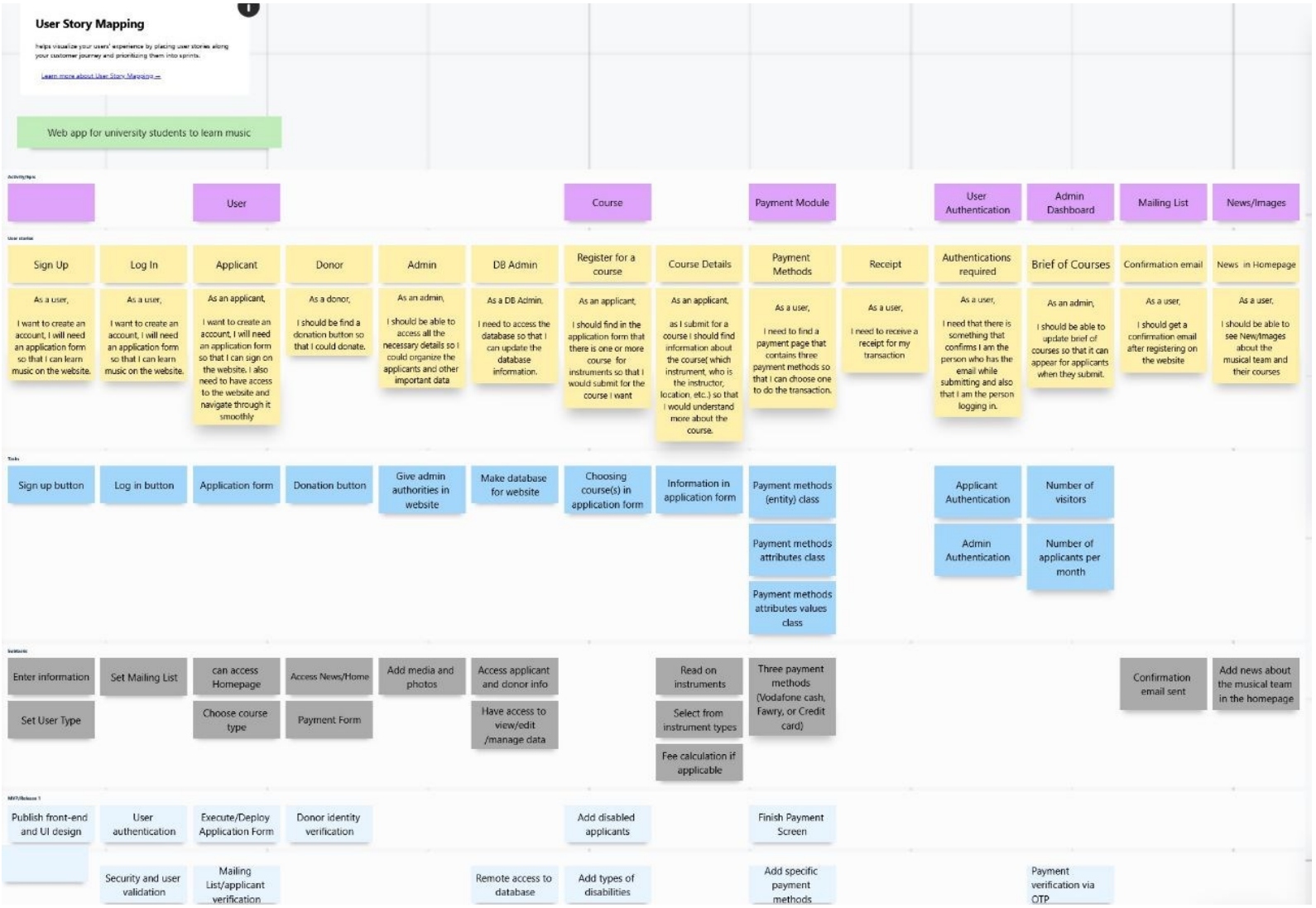
1- MODEL: The model component stores data and its related logic. It represents data that is being transferred between controller components or any other related business logic. It responds to the request from the views and also responds to instructions from the controller to update itself. It is also the lowest level of the pattern which is responsible for maintaining data. **In our project the model represents the data that is stored in the database as well as temporary data. Our models are divided into ASP.NET models and PHP Scripts.** (See code zip)

2- VIEW: A View is that part of the application that represents the presentation of data. Views are created by the data collected from the model data. A view requests the model to give information so that it presents the output presentation to the user. The view also represents the data from charts, diagrams, and tables. **In our project the view carries the user interface and direct creation of tuples in the database. It also has c# scripts that enable User Authentication and Admin Dashboard modules.**(See code zip)

3- CONTROLLER: The Controller is that part of the application that handles the user interaction. The controller interprets the mouse and keyboard inputs from the user, informing model and the view to change as appropriate. **In our project, the controllers allow user to call view methods that then communicate with data in model to allow user to directly enter their data in the database.** (See code zip)

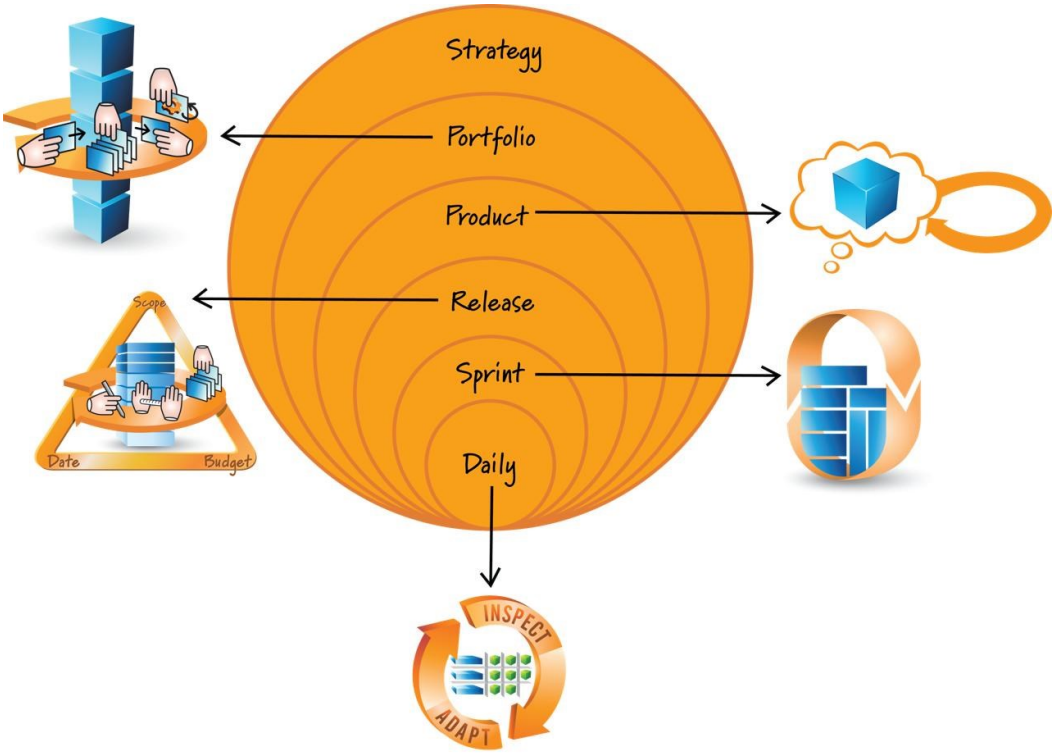
JIRA ACCOUNT

Story Map



MULTILEVEL PLANNING

On Scrum projects, teams plan at multiple levels of detail and at multiple times throughout product development. Formally Scrum defines only sprint planning and daily planning (via the daily scrum). However, most organizations also benefit from portfolio planning, product planning, and release planning. The different levels of planning on Scrum and Agile projects are illustrated in the picture that follows.



Our Application

Customer is to refer to a facebook page, help us to set priorities , ensure we are working toward common targets, creating a Strategic Plan that involves the implementation and evaluation of the results with regard to the organization's overall long-term goals and expectations.

HOME NEWS EVENTS VIDEOS APPLY HERE ADMIN ONLY LOGIN

Application Form

First Name

Middle Name

Last Name

Email

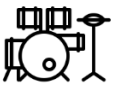
Phone Number

mm/dd/yyyy

Choose your Instrument

Created by André Luiz Gollo from the Noun Project

Created by André Luiz Gollo from the Noun Project



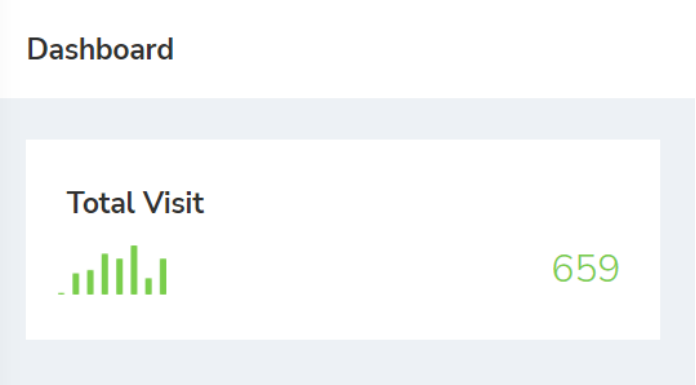
HOME NEWS EVENTS VIDEOS APPLY HERE ADMIN ONLY LOGIN

Sign In

Email

Password

LOG IN



Agile Portfolio Planning

The goal of agile portfolio planning is to determine which products to work on, in what order, and for how long. Agile portfolio planning is an activity involving stakeholders and product owners and tends to focus on a horizon of a year or more.

Agile Product Planning

Agile product planning, also called envisioning, is intended to capture the essence of a potential product and to create a rough plan for creating that product. The product owners and stakeholders who gather to plan new products tend to focus on a horizon of many months or longer. The participants consider the vision for a particular product and the evolution of the product over time. The primary outputs for envisioning are a vision, a high-level product backlog, and a product roadmap. These outputs become inputs for the higher-level portfolio planning.

When we are plan for a product we align on the outlines of our project that user want to reserve a course for an musical instrument and know everything about these instrument ,fees and schedule.

Release Planning in Scrum

In Scrum, release planning is about balancing customer value and overall quality against the constraints of scope, date, and budget for incremental deliveries. A release is typically planned for three to nine months in the future, though for some organizations a release might be even sooner.

Release planning in Scrum should involve the entire Scrum team (development team, ScrumMaster, product owner) as well as stakeholders. Most Scrum

development efforts do release planning after envisioning (product planning) and before the first sprint begins.
Now we start to plan for release our project and we align to release part after part by list we write it (login page,db scheme,course ,payment page,user)

Sprint Planning in Scrum

In Scrum, sprint planning happens at the beginning of each sprint. The entire Scrum team meets to determine the specific set of product backlog items that the team will accomplish in the next sprint. The outputs of sprint planning include a sprint goal and a sprint backlog: a description of the task-level work that has to be completed to get the product backlog items done.
Here we make a meeting before start a new sprint to ensure who made what to complete our goal without any confusion or 2 developer make same thing.

Daily Planning in Scrum

The most detailed level of planning occurs during the team's daily scrum meeting. This is the activity where the team members get together, and each person takes a turn sharing yesterday's accomplishments, today's plans, and any obstacles. The daily scrum is where team members collectively describe, in a highly visible way, the big-picture plan for that day.
At the end of everyday we make a zoom meeting every one make a thing tell us any issue he has and what he had done and we sure everything is work in the right way .

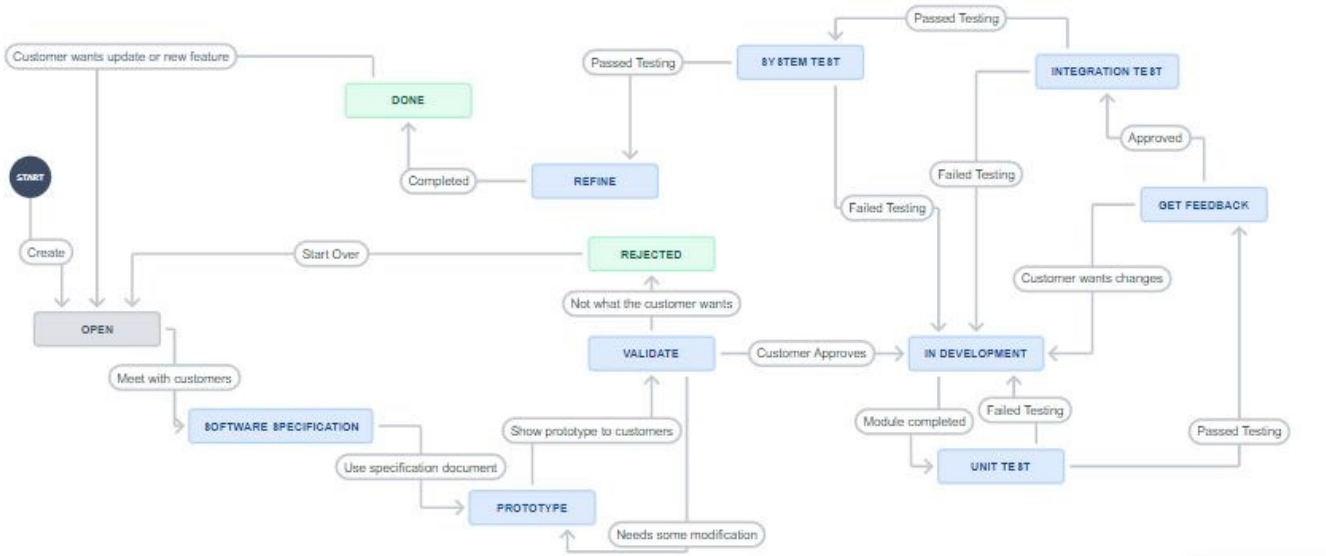
GOOD USER STORIES

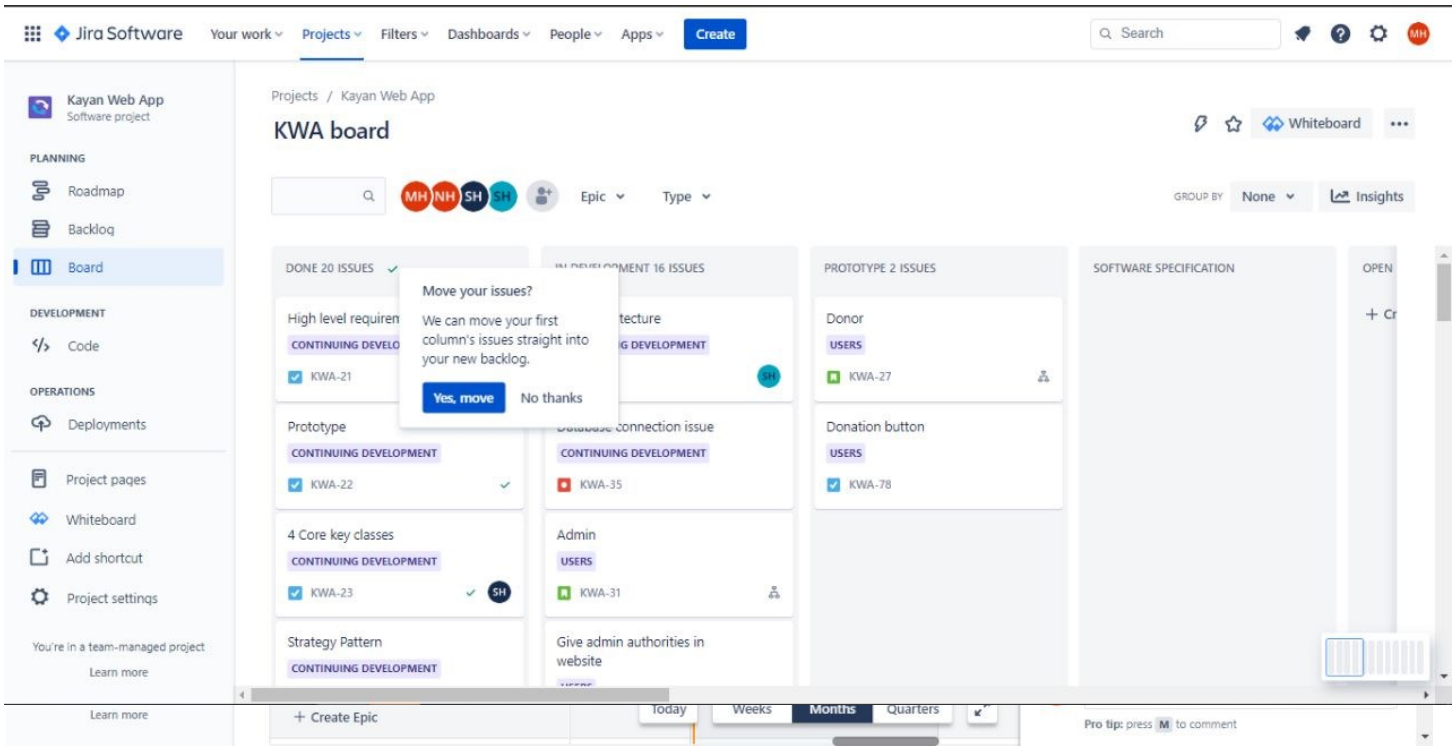
- Sign up: I want to create an account, I will need an application form so that I can sing on the website.
- Payment method: I need to find a payment page that contains three payment methods so that I can choose one to do the transaction , I should

be able to do the payment either with fawry, with credit/debt card or on site. I should be able to get a receipt of the payment containing details about my payment.

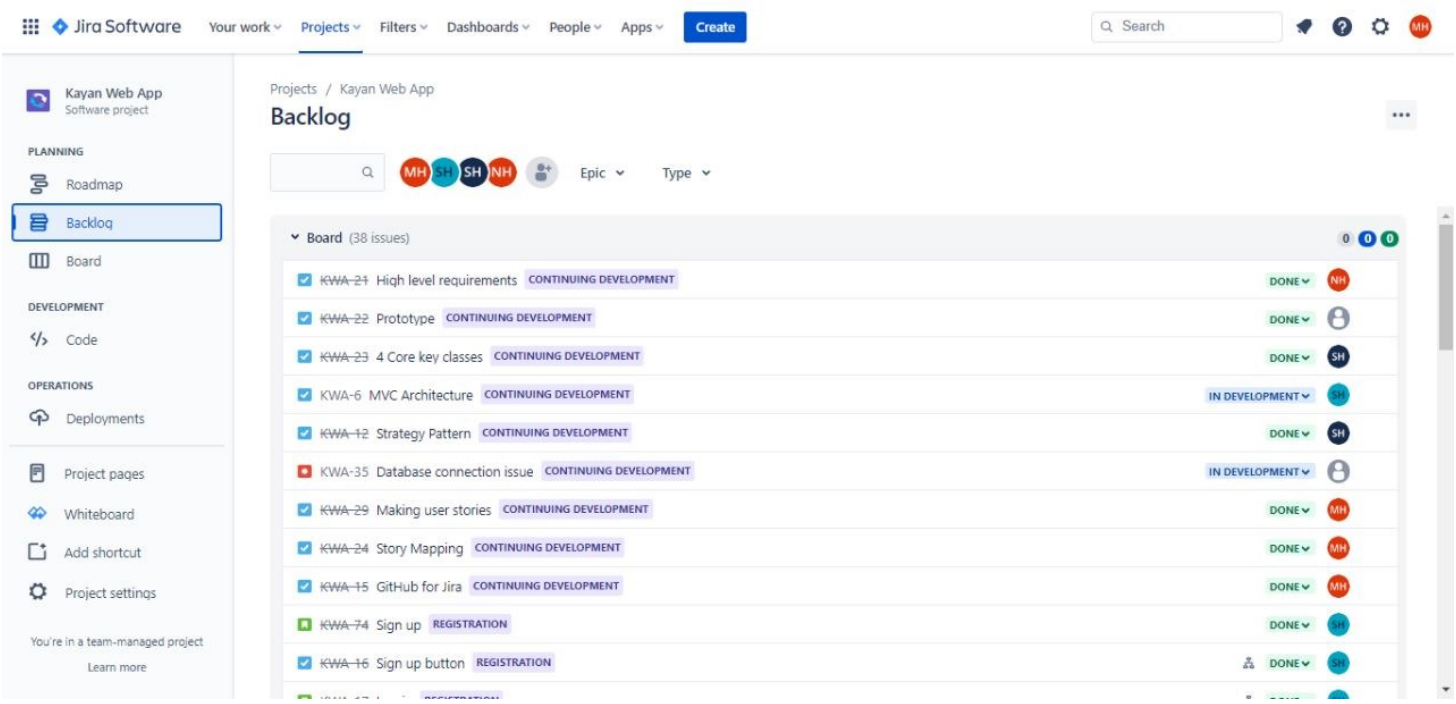
- Course details: As I submit for a course I should find information about the course(which instrument, who is the instructor, location, etc..) so that I would understand more about the course.

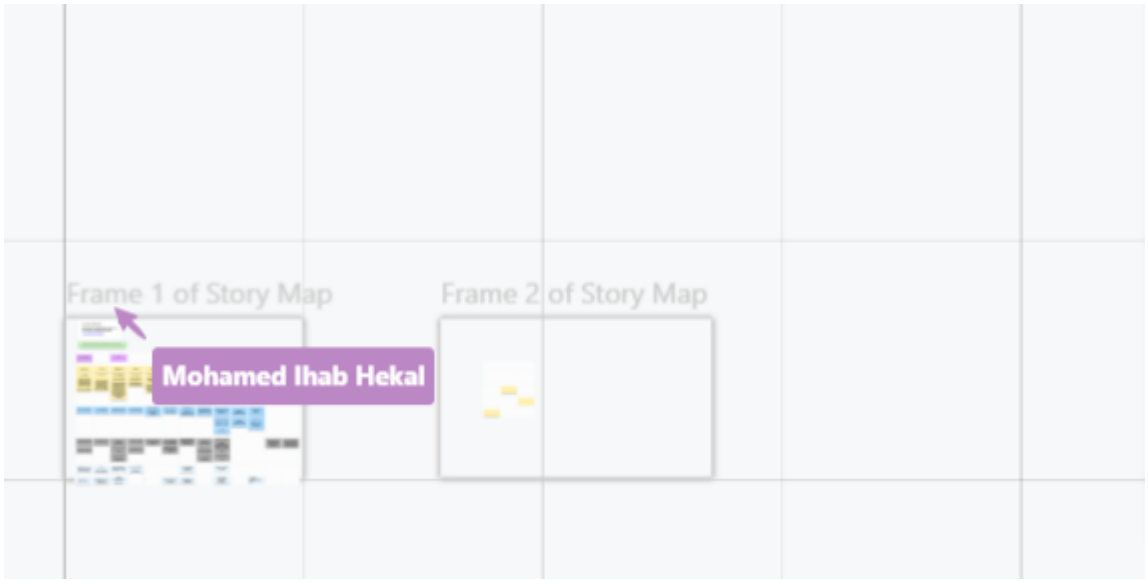
WORKFLOW





JIRA ACCOUNT





CLEANING CODE

Version 1 of kayan_demo

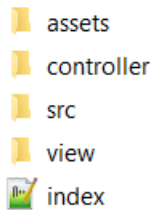
- Procedural, functional signup only. Code is not layered MVC.

connect.php

```php
if ( ! preg_match("/[a-z]/i", $_POST["password"])) {
    die("Password must contain at least one letter");
}

if ( ! preg_match("/[0-9]/", $_POST["password"])) {
    die("Password must contain at least one number");
}

if ($_POST["password"] !== $_POST["password_confirmation"]) {
    die("Passwords must match");
}

$password_hash = password_hash($_POST["password"], PASSWORD_DEFAULT);


try {
  $conn = new PDO("mysql:host=$servername;dbname=$dbname", $username, $password);
  // set the PDO error mode to exception
  $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);

  // begin the transaction
  $conn->beginTransaction();
  // our SQL statements
  $conn->exec("INSERT INTO `user` (userID, user_role)
  VALUES ('', '4');");

  $stmt = $conn->prepare("INSERT INTO `applicant`(applicantID, FullName, password, Email)
  VALUES (:id, :name, :pass, :email);");
  $id = $conn->lastInsertId();
  $stmt->bindParam(':id', $id);
  $stmt->bindParam(':name',  $_POST["name"]);
  $stmt->bindParam(':email', $_POST["email"]);
  $stmt->bindParam(':pass', $password_hash);
  $stmt->execute();
```

Version 2 of kayan_demo

- Using MVC layered architecture. Controller classes for each html file. HTML is in ‘views’, separate from PHP functions.



nect.php | sign-in.php

```php
<?php

class SignInController extends Controller {
    function defaultAction(){
        include 'view/sign-in.html';
    }

    function submitContactFormAction(){

        //validate
        //store data
        //send email

        echo 'Signed In Successfully';
    }
}
```

controller.php

```php
<?php

class Controller {
    //put your code here

    function runAction($actionName) {
        $actionName .= 'Action';
        if (method_exists($this, $actionName)){
            $this->$actionName();
        } else {
            echo "Error 404";
        }

    }
}
```

sign-in.html

```
                    </div>
                             
                    <img class="qanun" src="view/Images/qanun.jpg" alt="qanun"
                        style="display:inline-block;" />
                    <img class="drums"src="view/Images/drums.jpg" alt="drums"
                        style="display:inline-block;" />
                    <img class="mic" src="view/Images/mic.jpg" alt="mic"
                        style="display:inline-block;" />
                    <img class="violin" src="view/Images/violin.jpg" alt="violin"
                        style="display:inline-block;" />
                    <img class="piano" src="view/Images/piano.jpg" alt="piano"
                        style="display:inline-block;" />
                </div>
            </header>
        </head>
        <body>
            <h1>
                Sign In
            </h1>
            <div style="margin-top:-25px;">
            <form action="connect.php" method="post">
                    <label for="email"><b>Email</b></label>
                    <input type="text" placeholder="Enter Email" name="email" required>

                    <label for="psw"><b>Password</b></label>
                    <input type="password" placeholder="Enter Password" name="psw" required>

                    <button type="submit">Sign In</button>
                    <label>
                      <input type="checkbox" checked="checked" name="remember"> Remember me
                    </label>

                  <div style="background-color:#f1f1f1 ; margin-top: 10px;">
                    <button type="button" class="cancelbtn">Cancel</button>
                    <span class="psw">Forgot <a href="#">password?</a></span>
                  </div>
            </form>
        </div>
        </body>
        <footer style="background-color:black;">
            <div class="box"></div>
        </footer>
    </html>
```

- assets
    -- *css and js files imported from bootstrap*
- controller
    -- apply.php
    -- contact-us.php
    -- home.php
    -- sign-in.php
- src
    -- controller.php
- view
    -- *Images folder*
    -- apply.html
    -- contact-us.html
    -- home.html
    -- sign-in.html
- index.php

Final Version on Github

Link: https://github.com/ihabsalma12/CSE233-Agile