

**Happy Programming**  Requirement & Design Specification

**HappyProgramming System (HPS)**

**Version: 1.2**

– Hanoi, Nov 2024 –

A.Record of Changes

| **Version** | **Date** | **A\* M, D** | **In charge** | **Change Description** |
| --- | --- | --- | --- | --- |
| V0.1 | 5/9 | A | AnhNNH | Create RDS documents |
| V0.2 | 14/9 | A | KhanhNN, HuyMT, AnhNNH | Add Use Case diagram |
| V0.3 | 18/9 | A | All | Complete Iter1, add Use case Iter1 |
| V0.5 | 20/9 | A | KhanhNN | Add Context diagram, Add Screen Authorization |
| V0.6 | 1/10 | A | All | Complete Iter2, add Use case Iter2 |
| V0.7 | 5/10 | U | KhanhNN | Update Context diagram |
| V0.8 | 15/10 | A | All | Complete Iter3, add Use case Iter3 |
| V0.9 | 3/11 | A,U | KhanhNN | Update Context diagram, Update Actors, Add screen flow, Update Screen Authorization, Update Database Schema |
| V1.0 | 4/11 | A,U | KhanhNN, DatXN | Update Use Case diagram, Add Use Case diagram descriptions, screen flow descriptions, Update sequence number for UC |
| V1.1 | 6/11 | U | KhanhNN | Update Use case diagram descriptions, Update Table Descriptions, Update Context diagram |
| V1.2 | 08/11 | U | All | Update Use Case add business rule and error message |
|  |  |  |  |  |

\*A - Added M - Modified D - Deleted

B.Contents

***A.Record of Changes 2***

***B.Contents 3***  
**I. Overview 5**

[**1. System Overview 5**](#_heading=h.s6fmet4mw9p9)

[**2. Context diagram 5**](#_heading=h.zdrn79vbaf83)

[*1.1 Actors 6*](#_heading=h.3znysh7)

[*1.2 Use Cases Diagram 7*](#_heading=h.2et92p0)

[*2.3 Screens Flow 12*](#_heading=h.3dy6vkm)

[*2.2 Screen Descriptions 12*](#_heading=h.1t3h5sf)

[*2.3 Screen Authorization 14*](#_heading=h.4d34og8)

[**3. System High Level Design 16**](#_heading=h.17dp8vu)

[*3.1 Database Design 16*](#_heading=h.3rdcrjn)

[*3.2 Code Packages 23*](#_heading=h.35nkun2)

[**II. Requirement Specifications 24**](#_heading=h.1ksv4uv)

[**1. Public Functions 24**](#_heading=h.z337ya)

[*1.1 Signup System 24*](#_heading=h.3j2qqm3)

[*1.2 Sign In 25*](#_heading=h.74x6ix5sxi7k)

[*1.3 View all skill 26*](#_heading=h.1ybmm5mfenws)

[*1.4 View Profile-CV of mentor 27*](#_heading=h.qyf8yt5fc0mu)

[*1.5 Home Page 28*](#_heading=h.982frne7tjsh)

[**2.Mentee 28**](#_heading=h.3z5fsd7n9p42)

[*2.1 Create Request 28*](#_heading=h.789xaaqubkkw)

[*2.2 List Mentor suggestion 29*](#_heading=h.uat4gaqny0l6)

[*2.3 Payment 30*](#_heading=h.vj6jcx6mkux9)

[*2.4 Update request 31*](#_heading=h.siy85b9osbas)

[*2.5 List request by me 32*](#_heading=h.bknsryv4szet)

[*2.6 Statistic request by me 32*](#_heading=h.m3hbovjidwnu)

[*2.7 Rate & Comment 33*](#_heading=h.xulwv99ib1zb)

[*2.8 Update Mentee Profile 34*](#_heading=h.85uimfcazy5a)

[**3. Mentor 34**](#_heading=h.3cgxcnr3zsxg)

[*3.1 Create CV of mentor 34*](#_heading=h.225t5i1wguh5)

[*3.2 Update CV of mentor 35*](#_heading=h.ulsnzro9pbv5)

[*3.3 View statistic request 36*](#_heading=h.i4d49431ciap)

[*3.4 List of inviting request 37*](#_heading=h.kgl1kdcybje2)

[*3.5 CRUD Slot of Mentor 38*](#_heading=h.s4mayof8ln5)

[*3.6 Attendance Schedule 39*](#_heading=h.okzzhzhwwpot)

[**4. Admin 40**](#_heading=h.iwhfi4cfy7i7)

[*4.1 Admin add new Skill 40*](#_heading=h.yofnmocpsmw8)

[*4.2 Admin update Skill 41*](#_heading=h.bkm5gp3h2aax)

[*4.3 Admin view list of skill 43*](#_heading=h.clbksxkywdek)

[*4.4 View list of mentor 44*](#_heading=h.6nv13ztayarc)

[*4.5 View list of mentee 45*](#_heading=h.btaudz6my28r)

[*4.6 View all request of all mentee 46*](#_heading=h.i7109qj8p71s)

[*4.7 Admin dashboard 47*](#_heading=h.x53iziyuaqy9)

[**5. Manager 48**](#_heading=h.hjvk374cza0y)

[*5.1 Manager CV & Slot of mentor 48*](#_heading=h.nrux8uv7claw)

[*5.2 Manger payment request 49*](#_heading=h.bltzdlwf4ex7)

[**6. Common 51**](#_heading=h.75xpkgq10mh9)

[*6.1 View Slot 51*](#_heading=h.n0vh9mrqmc5m)

[*6.2 Authorization 51*](#_heading=h.g4hzcs7ymbx6)

[*6.3 Reset Password 52*](#_heading=h.e1ovv9ucocbu)

[*6.4 Change Password 52*](#_heading=h.8j4zfe7nkkqa)

[**III. Design Specifications 54**](#_heading=h.1ci93xb)

[**IV. Non-Functional Requirement 115**](#_heading=h.3as4poj)

[**V. Appendix 116**](#_heading=h.g7mbii3qgrbc)

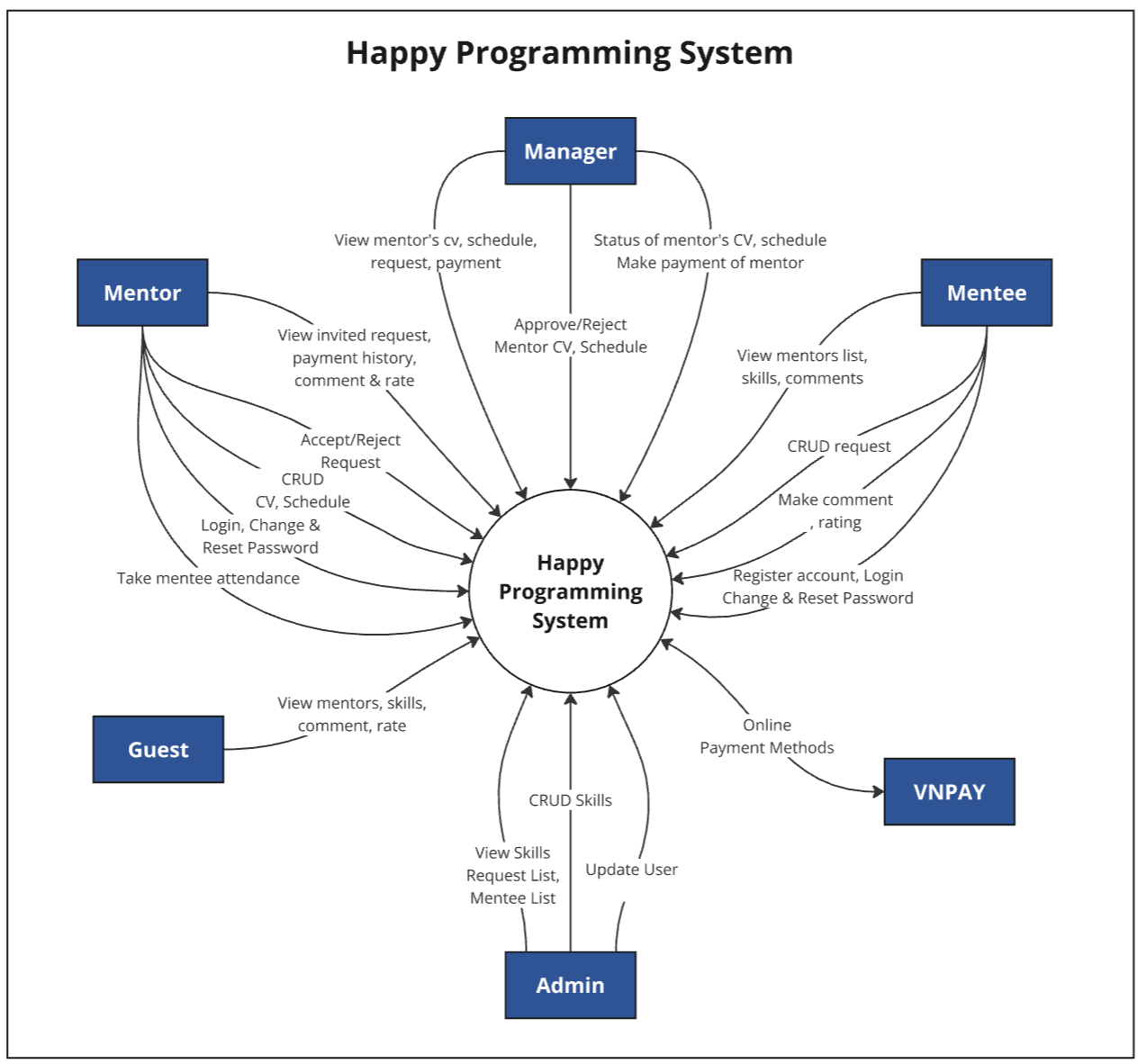
[*1. Common Error Definition 116*](#_heading=h.49x2ik5)

# I. Overview

## 1. System Overview

The **Happy Programming System** is a new software platform designed to facilitate online connections between mentors, mentees, and other actors involved in programming education. This system replaces the current manual processes of coordinating mentorship and learning activities. The context diagram below highlights the external entities and system interfaces for release 1.0. The Happy Programming System is envisioned to evolve over several releases, eventually integrating with online learning platforms, programming resources, and potentially third-party educational tools. The goal is to streamline the support and learning experience for mentees, allowing them to easily access mentorship and resources to enhance their programming skills.

## 2. Context diagram

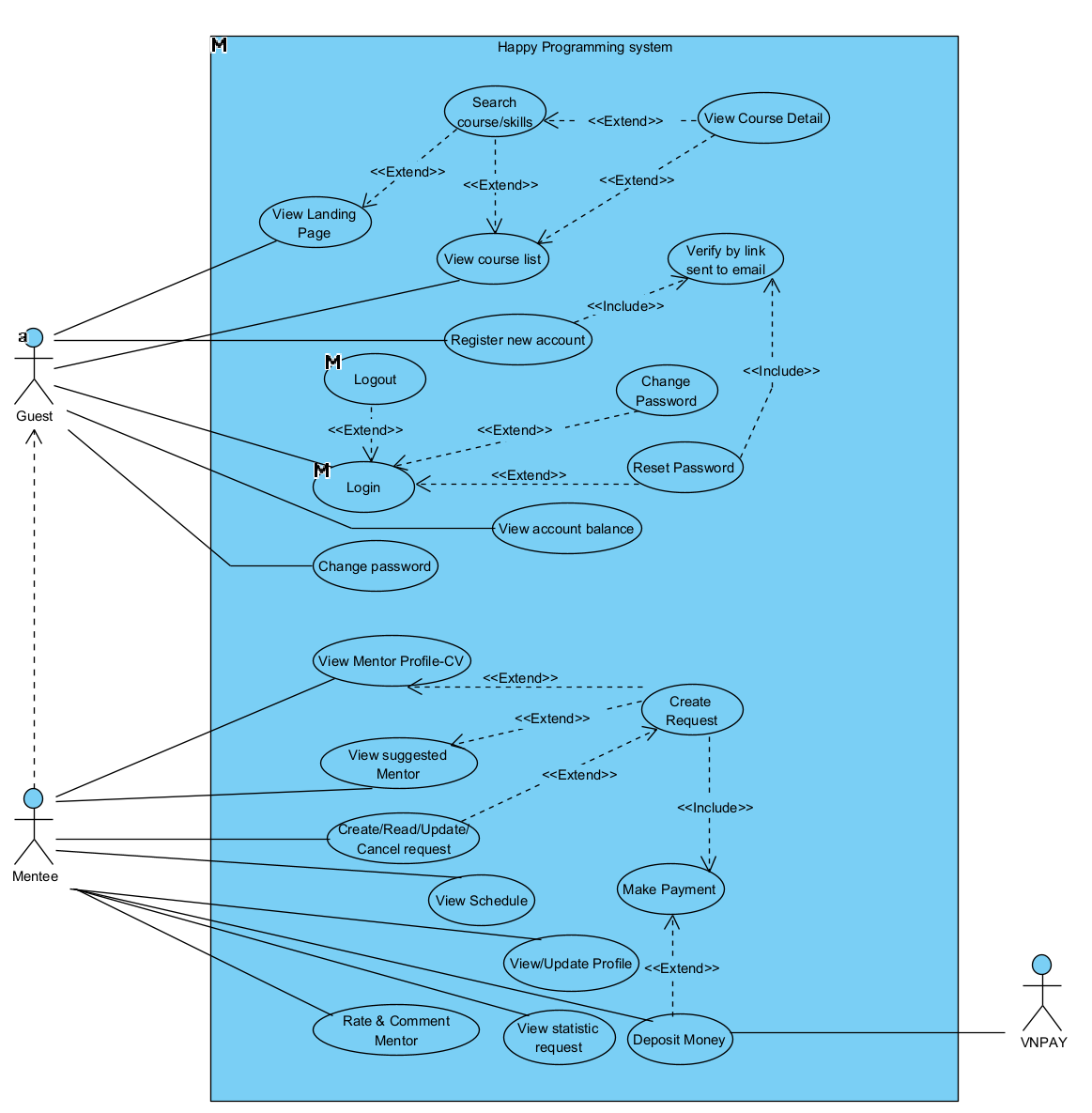


### 1.1 Actors

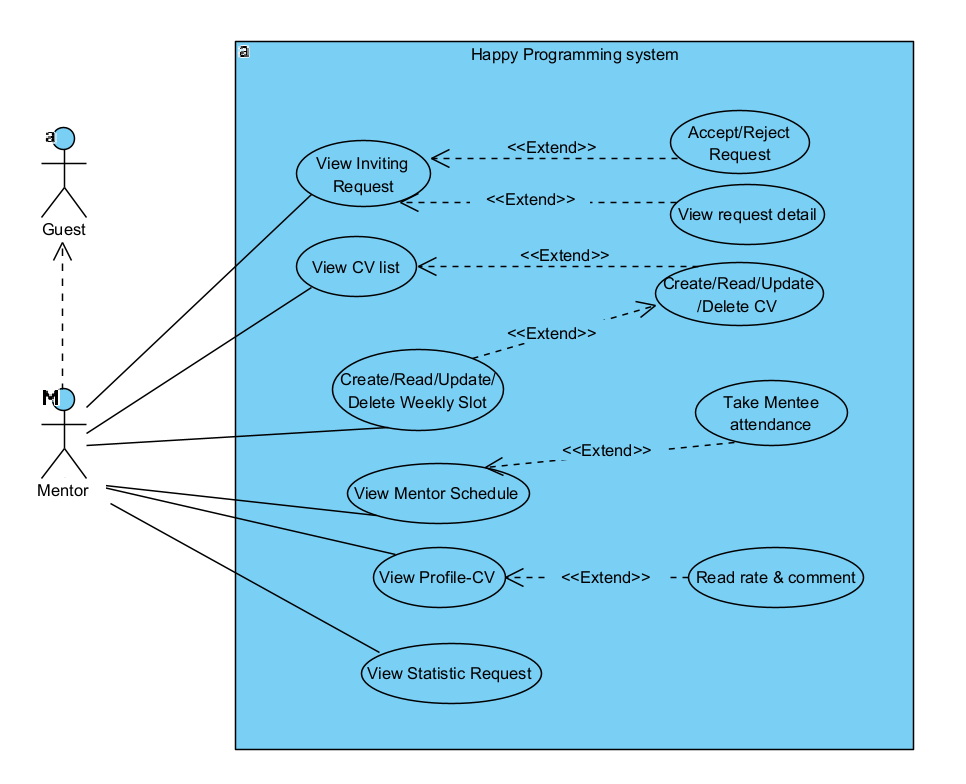
| **#** | **Actor** | **Description** |
| --- | --- | --- |
| 1 | Guest | The Guest role is intended for users who are not officially hiring a mentor or enrolled in a course in the Happy Programming system. Guests have limited access and functionality compared to registered users. |
| 2 | Admin | The Administrator role is the highest level of access and control in a Happy Programming system. Administrators have comprehensive privileges to manage all skills and users. |
| 3 | Manager | The Manager role is for users who oversee account management and provide accounts to admins. Managers have access to the right features and resources to facilitate their learning and management. |
| 4 | Mentor | The Mentor role is for experienced users who provide personalized guidance, support, and coaching to mentees within the happy programming system. Mentors leverage their knowledge and expertise to facilitate the growth and development of their mentees. |
| 5 | Mentee | The Mentee role is for users who are taking a mentoring course or receiving guidance. Mentees have access to tailored features and resources to facilitate their learning and development. |
| 6 | VNPAY | VNPAY's role is to provide online payment methods for customers when registering for a course on the happy programming system. Guests have access and payment within the system's checkout time. |

### 1.2 Use Cases Diagram

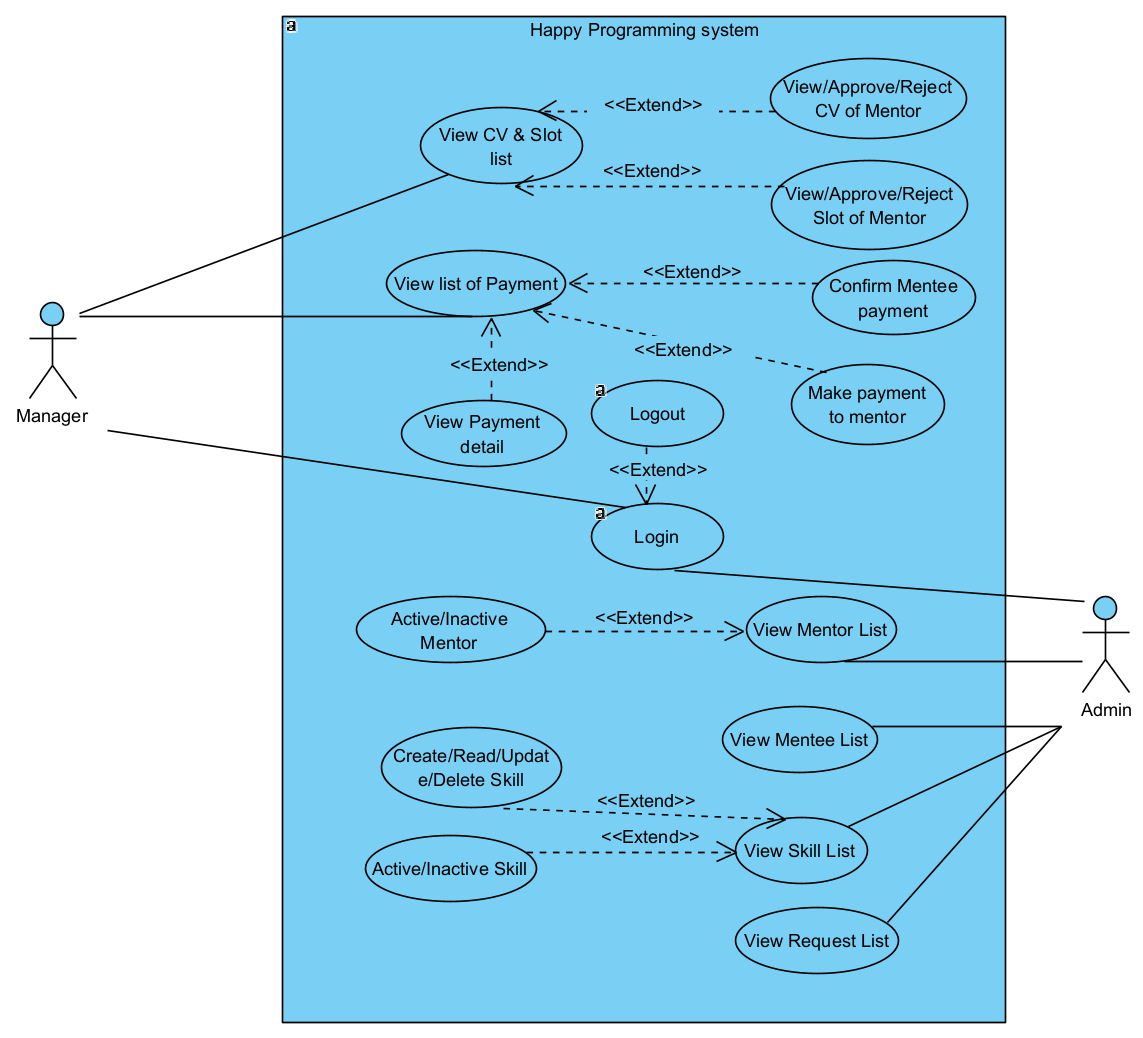
#### a. UCs for Guest and Mentee



#### b. UCs for Mentor



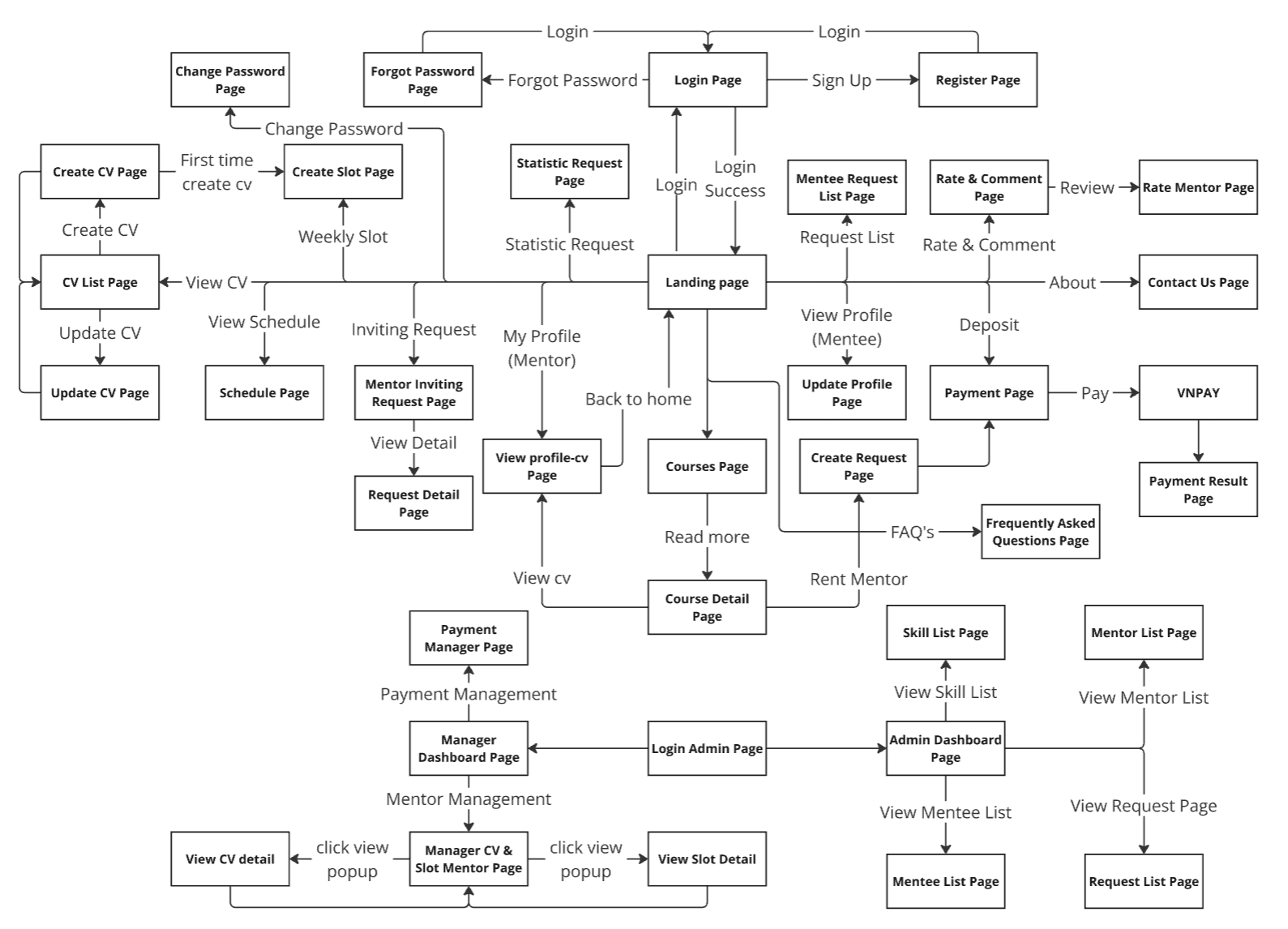
#### c. UCs for Backend



#### d. Descriptions

| **ID** | **Feature** | **Use Case** | **Use Case Description** |
| --- | --- | --- | --- |
| UC-1 | Signup System | Register new account | Allows users to create a new account by providing required information. |
| UC-2 | Login System | Login | Enables users to securely log into the system. |
| UC-3 | Reset Password | Reset Password | This screen allows users to enter their registered email to have a code sent to their email. |
| UC-4 | Change Password | Change Password | Enables users to change their password for account security. |
| UC-5 | Update Mentee Profile | View/Update Profile | Allows Mentees to view and update their personal profile information. |
| UC-6 | Rate & Comment Mentor | Rate & Comment Mentor | Provides a way for Mentees to rate and comment on Mentors after course. |
| UC-7 | View all skills | View course list | Allows users to view the full list of available skills/courses. |
| UC-8 | View profile-CV of Mentor | View Mentor Profile-CV | Enables Mentees to view the profile and CV of Mentors. |
| UC-9 | Home Page | View Landing Page | Displays the landing page with introductory and key content. |
| UC-10 | Create Request | Create Request | Allows Mentees to create requests for learning or mentorship. |
| UC-11 | List Mentor suggestion | View suggested Mentor | Shows suggested Mentors based on by the most request. |
| UC-12 | Create payment using VNPAY | Make Payment | Enables mentees to make payments through VNPAY. |
| UC-13 | Update request | Create/Read/Update/Cancel request | Provides full CRUD functionality for Mentees to manage their requests. |
| UC-14 | List request by me | Create/Read/Update/Cancel request | Shows a list of all requests created by the Mentee. |
| UC-15 | Statistic request by me | View statistic request | Displays statistics for the requests created by the Mentee. |
| UC-16 | Create CV of mentor | Create/Read/Update/Delete CV | Allows Mentors to create and manage their CVs to showcase their information. |
| UC-17 | Update CV of mentor | Create/Read/Update/Delete CV | Enables Mentors to update their CV with new information. |
| UC-18 | View statistic request | View Statistic Request | Shows overall statistics requests of mentors, helping track mentorship demand. |
| UC-19 | List of inviting request | View Inviting Request | Allows viewing requests of mentors. |
| UC-20 | Add new Skill | Create/Read/Update/Delete Skill | Allows Admin to add new skills mentor can teach. |
| UC-21 | Update Skill Information | Create/Read/Update/Delete Skill | Enables Admin to update information on the skills they offer. |
| UC-22 | Admin view list of skill | View Skill List | Allows Admins to view the complete list of skills available in the system. |
| UC-23 | View list of Mentor | View Mentor List | Displays a list of all Mentors registered in the system for admin. |
| UC-24 | View list of Mentee | View Mentee List | Shows a list of all Mentees registered in the system for admin. |
| UC-25 | View all requests of all Mentee | View Request List | Allows admin viewing of all requests submitted by Mentees. |
| UC-26 | Manager CV & Slot of mentor | View CV & Slot list | Provides a way to manage and view CVs and available slots of Mentors for managers. |
| UC-27 | Manager payment request | View list of Payment | Enables manager viewing and managing payment requests for requests. |
| UC-28 | View Slot of Mentee | View Schedule | Displays the schedule for Mentees, including upcoming and past sessions. |
| UC-29 | Authorization | Happy Programming system | Authorization and access control throughout the system. |
| UC-30 | CRUD Slot of mentor | Create/Read/Update/Delete Weekly Slot | Enables Mentors to create, update information on the slot they offer. |

### 2.3 Screens Flow



### 2.2 Screen Descriptions

| **#** | **Screen** | **Feature** | **Description** |
| --- | --- | --- | --- |
| 1 | Login Page | Authentication | This screen allows users to login to their account into the  system |
| 2 | Register Page | Authentication | Allows users to create a new account by providing required information. |
| 3 | Forgot Password Page | Authentication | This screen allows users to enter their registered email to have a code sent to their email. |
| 4 | Change Password Page | Authentication | Allows users to reset their password if they forget it. |
| 5 | Landing Page | Home | This screen is the website's home page, displaying courses, instructors, and course reviews. |
| 6 | Statistic Request Page | Statistics | This screen displays various statistical data related to the user's activity or requests. |
| 7 | Mentee Request List Page | Requests | This screen displays a list of requests made by mentees for course or request. |
| 8 | Rate & Comment Page | Feedback | This screen allows users to rate and comment on courses or instructors. |
| 9 | Contact Us Page | Support | This screen provides a form for users to reach out to the website's support team. |
| 10 | VNPAY Page | Payment | This screen redirects to the VNPAY payment portal for transactions. |
| 11 | Payment Result Page | Payment | This screen displays the result of a payment transaction. |
| 12 | Update Profile Page | Mentee Profile | This screen allows mentees to view and edit their profile information. |
| 13 | Create Request Page | Requests | This screen allows mentees to create requests for mentorship or specific sessions. |
| 14 | Courses Page | Courses | This screen displays a list of available courses for users to browse. |
| 15 | Course Detail Page | Courses | This screen displays detailed information about a specific course. |
| 16 | Frequently Asked Questions Page | Support | This screen displays frequently asked questions and answers for user guidance. |
| 17 | Payment Page | Payment | This screen allows users to proceed with payments for services or courses. |
| 18 | My Profile (Mentor) | Mentor Profile | This screen displays the mentor’s profile information. |
| 19 | View Profile-CV Page | Mentor Profile | This screen displays the CV or profile information of a mentor. |
| 20 | Create CV Page | Mentor Profile | This screen allows mentors to create or edit their CV information. |
| 21 | CV List Page | Mentor Profile | This screen displays a list of CVs for mentors in the system. |
| 22 | Update CV Page | Mentor Profile | This screen allows mentors to update or edit their CV information. |
| 23 | Schedule Page | Scheduling | This screen displays the mentor's or mentee's schedule for sessions. |
| 24 | Create Slot Page | Scheduling | This screen allows mentors to create available time slots for mentoring. |
| 25 | Mentor Inviting Request Page | Requests | This screen displays invitations for mentors to participate in requests. |
| 26 | Request Detail Page | Requests | This screen provides detailed information on a specific request made by a mentee. |
| 27 | Payment Manager Page | Payment | This screen allows the manager to view and manage all payment transactions. |
| 28 | Manager Dashboard Page | Management | This screen is the dashboard for managers to monitor cv slot and requests. |
| 29 | Manager CV & Slot Mentor Page | Management | This screen allows managers to view and manage status of mentor CVs and available time slots. |
| 30 | View CV Detail Page | Mentor Profile | This screen shows detailed CV information for a selected mentor. |
| 31 | View Slot Detail Page | Scheduling | This screen provides details on a specific time slot available for a selected mentor. |
| 32 | Login Admin Page | Authentication | This screen allows administrators like admin and manager to log in to the system. |
| 33 | Admin Dashboard Page | Admin | This screen is the main dashboard for administrators to manage system-wide activities. |
| 34 | Skill List Page | Admin | This screen displays a list of skills available for the admin to edit from. |
| 35 | Mentor List Page | Admin | This screen displays a list of mentors to help admin manager status. |
| 36 | Mentee List Page | Admin | This screen displays a list of mentees to help the admin manager. |
| 37 | Request List Page | Admin | This screen displays a list of request to help admin manager status. |
| 38 | Statistic Request Page | Mentor | This screen provides statistical data specifically related to mentors. |

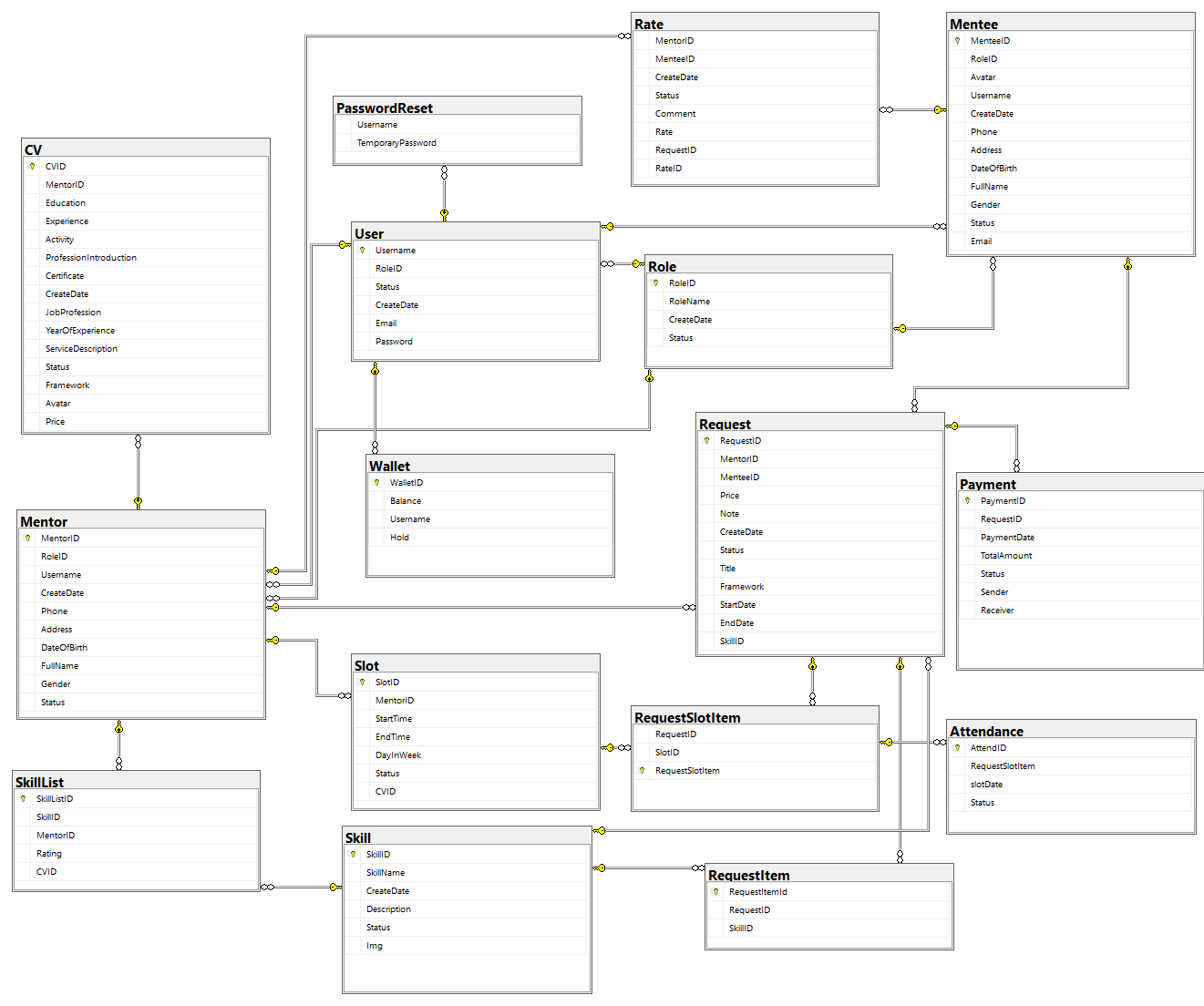
### 2.3 Screen Authorization

| **Screen** | **Mentor** | **Mentee** | **Manager** | **Admin** | **Guest** |
| --- | --- | --- | --- | --- | --- |
| Login Page | X | X |  |  | X |
| Register Page | X | X |  |  | X |
| Forgot Password Page | X | X |  |  | X |
| Landing Page | X | X |  |  | X |
| Statistic Request Page | X | X |  |  |  |
| Mentee Request List Page |  | X |  |  |  |
| Rate & Comment Page |  | X |  |  |  |
| Contact Us Page | X | X |  |  | X |
| VNPAY Page |  | X |  |  |  |
| Payment Result Page |  | X |  |  |  |
| Update Profile Page |  | X |  |  |  |
| Create Request Page |  | X |  |  |  |
| Courses Page | X | X |  |  | X |
| Course Detail Page | X | X |  |  | X |
| Frequently Asked Questions Page | X | X |  |  | X |
| Payment Page |  | X |  |  |  |
| My Profile (Mentor) | X |  |  |  |  |
| View Profile-CV Page | X | X |  |  | X |
| Create CV Page | X |  |  |  |  |
| CV List Page | X |  |  |  |  |
| Update CV Page | X |  |  |  |  |
| Schedule Page | X | X |  |  |  |
| Create Slot Page | X |  |  |  |  |
| Mentor Inviting Request Page | X |  |  |  |  |
| Request Detail Page | X | X |  |  |  |
| Payment Manager Page |  |  | X |  |  |
| Manager Dashboard Page |  |  | X |  |  |
| Manager CV & Slot Mentor Page |  |  | X |  |  |
| View CV Detail Page |  |  | X |  |  |
| View Slot Detail Page |  |  | X |  |  |
| Login Admin Page |  |  | X | X |  |
| Admin Dashboard Page |  |  |  | X |  |
| Skill List Page |  |  |  | X |  |
| Mentor List Page |  |  |  | X |  |
| Mentee List Page |  |  |  | X |  |
| Request List Page |  |  |  | X |  |
| Statistic Request Page | X | X |  |  | X |

## 3. System High Level Design

### 3.1 Database Design

#### a. Database Schema



#### b. Table Descriptions

##### 1. Role Table

| **#** | **Field name** | **Type** | **Nullable** | **PK/FK** | **Notes** |
| --- | --- | --- | --- | --- | --- |
| 1 | RoleID | Integer | N | PK | Auto-generated ID |
| 2 | RoleName | String | N |  | Role name (max 10 chars) |
| 3 | CreateDate | Date | N |  | Role creation date |
| 4 | Status | String | Y |  | Role status (max 20 chars) |

##### 2. User Table

| **#** | **Field name** | **Type** | **Nullable** | **PK/FK** | **Notes** |
| --- | --- | --- | --- | --- | --- |
| 1 | Username | String | N | PK | Username (max 30 chars) |
| 2 | RoleID | Integer | N | FK | Foreign key referencing Role.RoleID |
| 3 | Status | String | Y |  | User status (max 20 chars) |
| 4 | CreateDate | Date | N |  | User creation date |
| 5 | Email | String | N |  | User email (max 50 chars) |
| 6 | Password | String | N |  | User password (max 50 chars) |

##### 3. Mentor Table

| **#** | **Field name** | **Type** | **Nullable** | **PK/FK** | **Notes** |
| --- | --- | --- | --- | --- | --- |
| 1 | MentorID | Integer | N | PK | Auto-generated ID |
| 2 | RoleID | Integer | N | FK | Foreign key referencing Role.RoleID |
| 3 | Username | String | N | FK | Foreign key referencing User.Username |
| 4 | CreateDate | Date | Y |  | Mentor creation date |
| 5 | Phone | String | Y |  | Mentor phone (max 15 chars) |
| 6 | Address | String | Y |  | Mentor address (max chars: MAX) |
| 7 | DateOfBirth | Date | Y |  | Mentor date of birth |
| 8 | FullName | String | Y |  | Mentor full name (max 50 chars) |
| 9 | Gender | String | Y |  | Gender (max 10 chars) |
| 10 | Status | String | Y |  | Mentor status (max 20 chars) |

##### 4. Mentee Table

| **#** | **Field name** | **Type** | **Nullable** | **PK/FK** | **Notes** |
| --- | --- | --- | --- | --- | --- |
| 1 | MenteeID | Integer | N | PK | Auto-generated ID |
| 2 | RoleID | Integer | N | FK | Foreign key referencing Role.RoleID |
| 3 | Avatar | Binary | Y |  | Avatar image (VARBINARY(MAX)) |
| 4 | Username | String | N | FK | Foreign key referencing User.Username |
| 5 | CreateDate | Date | N |  | Mentee creation date |
| 6 | Email | String | N |  | Mentee email (max 50 chars) |
| 7 | Phone | String | Y |  | Mentee phone (max 15 chars) |
| 8 | Address | String | Y |  | Mentee address (max chars: MAX) |
| 9 | DateOfBirth | Date | Y |  | Mentee date of birth |
| 10 | FullName | String | N |  | Mentee full name (max 50 chars) |
| 11 | Gender | String | N |  | Gender (max 10 chars) |
| 12 | Status | String | Y |  | Mentee status (max 20 chars) |

##### 5. Skill Table

| **#** | **Field name** | **Type** | **Nullable** | **PK/FK** | **Notes** |
| --- | --- | --- | --- | --- | --- |
| 1 | SkillID | Integer | N | PK | Auto-generated ID |
| 2 | SkillName | String | N |  | Skill name (max 50 chars) |
| 3 | CreateDate | Date | N |  | Skill creation date |
| 4 | Description | String | Y |  | Skill description (max chars: MAX) |
| 5 | Status | String | Y |  | Skill status (max 20 chars) |
| 6 | Img | String | Y |  | Skill image (NVARCHAR(MAX)) |

##### 6. SkillList Table

| **#** | **Field name** | **Type** | **Nullable** | **PK/FK** | **Notes** |
| --- | --- | --- | --- | --- | --- |
| 1 | SkillListID | Integer | N | PK | Auto-generated ID |
| 2 | SkillID | Integer | N | FK | Foreign key referencing Skill.SkillID |
| 3 | MentorID | Integer | N | FK | Foreign key referencing Mentor.MentorID |
| 4 | Rating | Integer | Y |  | Skill rating |
| 5 | CVID | Integer | Y |  | CvId |

##### 7. Request Table

| **#** | **Field name** | **Type** | **Nullable** | **PK/FK** | **Notes** |
| --- | --- | --- | --- | --- | --- |
| 1 | RequestID | Integer | N | PK | Auto-generated ID |
| 2 | MentorID | Integer | N | FK | Foreign key referencing Mentor.MentorID |
| 3 | MenteeID | Integer | N | FK | Foreign key referencing Mentee.MenteeID |
| 4 | Price | Float | N |  | Price for the request |
| 5 | Note | String | Y |  | Request notes (max chars: MAX) |
| 6 | CreateDate | Date | N |  | Request creation date |
| 7 | Status | String | Y |  | Request status (max 20 chars) |
| 8 | Title | String | N |  | Request title (max 50 chars) |
| 9 | StartDate | Time | N |  | Start date |
| 10 | EndDate | Date | N |  | End date |
| 11 | Framework | String | N |  | Framework (max 30 chars) |
| 12 | SkillID | Integer | N |  |  |

##### 8. Payment Table

| **#** | **Field name** | **Type** | **Nullable** | **PK/FK** | **Notes** |
| --- | --- | --- | --- | --- | --- |
| 1 | PaymentID | Integer | N | PK | Auto-generated ID |
| 2 | RequestID | Integer | N | FK | Foreign key referencing Request.RequestID |
| 3 | PaymentDate | Date | N |  | Payment creation date |
| 4 | TotalAmount | Float | N |  | Total payment amount |
| 5 | PaymentDate | Date | N |  | Payment date |
| 6 | Status | String | Y |  | Payment status (max 20 chars) |
| 7 | Sender | String | Y |  | Username of sender (max 30 chars) |
| 8 | Receiver | String | N |  | Username of receiver (max 30 chars) |

##### 9. CV Table

| **#** | **Field name** | **Type** | **Nullable** | **PK/FK** | **Notes** |
| --- | --- | --- | --- | --- | --- |
| 1 | CVID | Integer | N | PK | Auto-generated ID |
| 2 | MentorID | Integer | N | FK | Foreign key referencing Mentor.MentorID |
| 3 | Education | String | N |  | Mentor's highest educational qualification |
| 4 | Experience | String | N |  | Detailed experience description |
| 5 | Activity | String | N |  | Mentor's activities |
| 6 | ProfessionIntroduction | String | N |  | Introduction to mentor's profession |
| 7 | Certificate | String | N |  | Relevant certificates |
| 8 | CreateDate | DateTime | N |  | Profile creation date |
| 9 | JobProfession | String | N |  | Mentor's job profession |
| 10 | YearOfExperience | Integer | N |  | Total years of experience |
| 11 | ServiceDescription | String | N |  | Description of services (max 30 chars) |
| 12 | Status | String | N |  | Current status (max 20 chars) |
| 13 | Framework | String | N |  | Preferred frameworks (max 30 chars) |
| 14 | Avatar | Binary | N |  | Binary data for mentor's profile image |
| 15 | Price | Float | N |  | Service price |

##### 10. Slot Table

| 1 | SlotID | Integer | N | PK | Auto-generated ID |
| --- | --- | --- | --- | --- | --- |
| 2 | MentorID | Integer | N | FK | Foreign key referencing Mentor.MentorID |
| 3 | StartTime | Time | N |  | Slot start time |
| 4 | EndTime | Time | N |  | Slot end time |
| 5 | DayInWeek | String | N |  | Day of the week (e.g., Monday, Tuesday) |
| 6 | Status | String | N |  | Slot status |
| 7 | CVID | Integer | N | FK | Foreign key referencing CV.CVID |

##### 11. RequestSlotItem Table

| 1 | RequestID | Integer | N | FK | Foreign key referencing Request.RequestID |
| --- | --- | --- | --- | --- | --- |
| 2 | SlotID | Integer | N | FK | Foreign key referencing Slot.SlotID |
| 3 | RequestSlotItem | Integer | N |  | Unique identifier for each request-slot mapping |

##### 12. Attendance Table

| 1 | AttendID | Integer | N | PK | Auto-generated ID for attendance record |
| --- | --- | --- | --- | --- | --- |
| 2 | RequestSlotItem | Integer | N | FK | Foreign key referencing RequestSlot.RequestSlotItem |
| 3 | SlotDate | DateTime | N |  | Date of the slot |
| 4 | Status | String | N |  | Attendance status (e.g., Present, Absent) |

##### 

##### 13. RequestItem Table

| 1 | RequestItemId | Integer | N | PK | Auto-generated ID for request item |
| --- | --- | --- | --- | --- | --- |
| 2 | RequestID | Integer | N | FK | Foreign key referencing Request.RequestID |
| 3 | SkillID | Integer | N | FK | Foreign key referencing Skill.SkillID |

##### 14. Wallet Table

| 1 | WalletID | Integer | N | PK | Auto-generated ID for each wallet |
| --- | --- | --- | --- | --- | --- |
| 2 | Balance | Float | N |  | Current balance in the wallet |
| 3 | Username | String | N |  | Username associated with the wallet |
| 4 | Hold | Float | N |  | Amount on hold (e.g., pending transactions) |

##### 15. PasswordReset Table

| 1 | Username | String | N | PK | Username associated with the temporary password |
| --- | --- | --- | --- | --- | --- |
| 2 | TemporaryPassword | String | N |  | Temporary password for user authentication |

##### 16. Rate Table

| 1 | RateID | Integer | N | PK | Auto-generated ID for each feedback record |
| --- | --- | --- | --- | --- | --- |
| 2 | MentorID | Integer | N | FK | Foreign key referencing Mentor.MentorID |
| 3 | MenteeID | Integer | N | FK | Foreign key referencing Mentee.MenteeID |
| 4 | CreateDate | Date | N |  | Date when the feedback was created |
| 5 | Status | String | N |  | Status of the feedback (e.g., Approved, Pending) |
| 6 | Comment | String | N |  | Feedback comment |
| 7 | Rate | Integer | N |  | Rating score (e.g., 1-5) |
| 8 | RequestID | Integer | N | FK | Foreign key referencing Request.RequestID |

### 3.2 Code Packages

| **No** | **Package** | **Description** |
| --- | --- | --- |
| 1 | DAO | Contains Data Access Object classes, providing methods to access and manipulate data in the database. |
| 2 | Model | Contains classes that represent data entities, corresponding to tables in the database. |
| 3 | controller | General controller classes responsible for managing data flow between the interface and business logic. |
| 4 | controller.Admin | Handles and processes functions related to the admin role, including administrative tasks. |
| 5 | controller.CV | Manages CV-related functions, such as viewing, creating, and updating CVs. |
| 6 | controller.Manager | Responsible for manager functions, such as reviewing and approving mentor CVs and Slots. |
| 7 | controller.Mentee | Manages mentee-related functionalities, including registration and profile management. |
| 8 | controller.Mentor | Handles mentor-related functionalities, such as profile CV updates and courses management. |
| 9 | Filter | Contains filter classes used to manage and control access based on specific criteria. |
| 10 | VnPay | Manages payment processing functionalities with the VnPay API integration. |
| 11 | util | Contains utility classes and methods that provide common functionalities used across the system. |

# II. Requirement Specifications

### 

## 1. Public Functions

### 1.1 Signup System

#### a. Functional Description

| UC ID and Name: | **UC-1 Signup System** | | |
| --- | --- | --- | --- |
| Created By: | AnhNHH | Date Created: | 18/09/2024 |
| Primary Actor: | Guest | Secondary Actors: | None |
| Trigger: | 1. User clicks Register link from the page header OR 2. User accesses an authenticated feature (from a link or type the page URL directly into the address bar) | | |
| Description: | As a user, I want to be able to create new account to the system so that I can use the system’s authenticated features. | | |
| Preconditions: | * A unique username is required. * A unique email is required. * Email must be in a valid format. * Password must meet specified format requirements. | | |
| Postconditions: | * User successfully signs up for an account. * A confirmation email is sent to the user’s email address. | | |
| Normal Flow | 1. User accesses the User Sign Up screen. 2. User enters the required sign-up details. 3. User clicks the "Sign Up" button. 4. System validates the sign-up details (refer to Exception E1). 5. System allows the user to sign up. 6. System redirects the user to the login page. | | |
| Alternative Flows: | 1. **Invalid email format:** If the user enters an invalid email format, the website displays an error message and prompts the user to correct it. 2. **Account already exists:** If the user tries to sign up with an email address that is already exist , the website displays an error message the user to choose a different email address. 3. **Invalid password format:** If password not meet the requirement, the website display the message that show the password requirement 4. **Password not match:** If password and repass is different show user the notification | | |
| Exceptions: | ***E1 System can’t reach database***  1. The Error 500 screen page is shown to the user | | |
| Priority: | Must Have | | |
| Frequency of Use: | High | | |
| Business Rules: | +User must provide a valid and unique email address.  +User must create a unique username (if applicable).  +User may have additional requirements for passwords | | |
| Other Information: |  | | |
| Assumptions: | Users have access to a reliable internet connection, understand the sign-up process, and can access their email for verification. | | |

### 1.2 Sign In

#### a. Functional Description

| UC ID and Name: | **UC-2 SignInSystem** | | |
| --- | --- | --- | --- |
| Created By: | AnhNHH | Date Created: | 20/09/2024 |
| Primary Actor: | Guest | Secondary Actors: | None |
| Trigger: | User clicks Register link from the page header, or  User accesses an authenticated feature (from a link or type the page URL directly into the address bar) | | |
| Description: | As a Mentee/Mentor/, I want to be able to sign in so that I can use the system’s authenticated features and access my personalized account. | | |
| Preconditions: | User account has been created & authorized | | |
| Postconditions: | The user successfully signs in and gains access to their account and personalized features. | | |
| Normal Flow | 1.1.User navigates to the login screen.  1.2.User enters their login credentials.  1.3.User clicks the "Sign In" button.  1.4.System validates the credentials (SN01)(SN02).  1.5.System grants access if credentials are correct.  1.6.System redirects the user to the homepage. | | |
| Alternative Flows: | 2.1.User clicks the "Forgot Password" link on the login screen.  2.2.System prompts the user to enter their registered email address.  2.3.System sends a password reset link to the user's email.  2.4.User follows the link and resets their password.  2.5.User returns to the login screen and enters their new password to sign in. | | |
| Exceptions: | ***1.4 Username or password not found(*** | | |
| Priority: | Must Have | | |
| Frequency of Use: | High | | |
| Business Rules: |  | | |
| Other Information: |  | | |
| Assumptions: | Users have previously registered or been authorized.  Users know their login credentials. | | |

#### b.Business rule

| **ID** | **Rule Definition** |
| --- | --- |
| SN01 | The system must validate the presence of both username and pass parameters. |
| SN02 | User must validate account through link in email |

#### c.Error message

| **ID** | **Error Message** |
| --- | --- |
| SNE01 | Username and pass is invaild |
| SNE02 | Account is not validate by email link |

### 1.3 View all skill

#### a. Functional Description

| **Name Of Function:** | **UC-7 View all skills** | | | |
| --- | --- | --- | --- | --- |
| **Created By:** | AnhNHH |  | **Created Date:** | 1/10/2024 |
| **Primary Actor:** | Public |  | **Secondary Actor:** |  |
| **Description:** | View all skills of HappyProgramming system and related information | | |  |
| **Trigger:** | An user wanna views all skills of HappyProgramming Academy | | | |
| **Preconditions:** | User/guest access website | | | |
| **Postconditions:** | User sees all skills, views skill details and mentors have that skill. | | | |
| **Normal Flow:** | 1. User enter home page 2. User access view all skill page 3. System display all skill order by skill name and has STT 4. User view skill detail | | | |
| **Alternative Flows:** | 2.2 User search  2.3 User click top course in home page  3.2 System display skill by search  3.3 System display all skill order by skill name and has STT  4.2 User view mentors have that skill | | | |
| **Exceptions:** | ***E1 System can’t reach database***  1. The Error 500 screen page is shown to the user | | |  |
| **Priority:** | High | | | |
| **Frequency of Use:** | High | | | |
| **Business Rules:** | N/A | | | |
| **Other Information:** | N/A | | | |
| **Assumptions:** | N/A | | | |

#### 

### 1.4 View Profile-CV of mentor

#### a.Functional Description

#### 

| UC ID and Name: | **UC-8\_View profile-cv of Mentor** | | |
| --- | --- | --- | --- |
| Created By: | Khanhnn | Date Created: | 6/10/2024 |
| Primary Actor: | Public | Secondary Actors: | None |
| Trigger: | **User clicks View profile-cv of Mentor button from the home page** | | |
| Description: | View profile-cv of Mentor | | |
| Preconditions: | 1. Mentor CV been active and exist | | |
| Postconditions: | 1. System successfully load Mentor profile-cv 2. System unsuccessfully load Mentor profile-cv and show error | | |
| Normal Flow | **View profile-cv of Mentor**  1. User enter mentor list  2. User click on mentor detail  3. System show profile-cv of Mentor, rating & comment | | |
| Alternative Flows: | 1.2 User enter comment info button from home page  2.2 System show profile-cv of Mentor, rating & comment | | |
| Exceptions: | 4. Avatar don't display(ER01)  6.System display fail(ER02) | | |
| Priority: | High | | |
| Frequency of Use: | High | | |
| Business Rules: | None | | |
| Other Information: | None | | |
| Assumptions: | None | | |

#### b.Business rule

| **ID** | **Rule Definition** |
| --- | --- |
| BR01 | Avatar file must smaller than 5mb |
| BR02 | Text must correct in format |

#### c.Error message

| **ID** | **Error Definition** |
| --- | --- |
| ER01 | Display default avatar |
| ER02 | 500 Database Error |

### 1.5 Home Page

#### a.Functional Description

#### 

| UC ID and Name: | **UC-9\_Home Page** | | |
| --- | --- | --- | --- |
| Created By: | Khanhnn | Date Created: | 6/10/2024 |
| Primary Actor: | Guest | Secondary Actors: | All |
| Trigger: | **Enter /happy\_programming/home** | | |
| Description: | View landing page | | |
| Preconditions: | 1. Skill and skillList must created 2. Rating data must created | | |
| Postconditions: | 1. System successfully load home screen 2. System unsuccessfully load home screen and show error | | |
| Normal Flow | **Home Page**  1. User enter home page by link or home button | | |
| Alternative Flows: | 1.2 Using navigation bar to get back home | | |
| Exceptions: | 1.Home data don't display (EM02) | | |
| Priority: | High | | |
| Frequency of Use: | Very High | | |
| Business Rules: | None | | |
| Other Information: | None | | |
| Assumptions: | None | | |

#### b.Business rule

| **ID** | **Rule Definition** |
| --- | --- |
| None | None |

#### c.Error message

| **ID** | **Error Message** |
| --- | --- |
| EM02 | 500 Database Error |

## 2.Mentee

### 2.1 Create Request

#### a. Functional Description

| **Name Of Function:** | **UC10\_Create Request** | | | |
| --- | --- | --- | --- | --- |
| **Created By:** | AnhNHHE |  | **Created Date:** | 5/10/2024 |
| **Primary Actor:** | User(Mentee) |  | **Secondary Actor:** | System |
| **Description:** | Create request to get support from mentor | | | |
| **Trigger:** | User click on “Rent mentor” button to request support from mentor | | | |
| **Preconditions:** | - The user(mentee) is signed in their account.  - The user(mentee) access create request page | | | |
| **Postconditions:** | - A new request is created and saved in the database.  - The request status is set to “Open" | | | |
| **Normal Flow:** | 1. Mentee login and accesses the skill list screen  2. Mentee click on skill they want to learn  3. Mentee choose mentor and click on “Rent mentor” button  4. Mentee enter required information and submit  5. The system validates input information  6. System create new request with status “Open” | | | |
| **Alternative Flows:** | None | | | |
| **Exceptions:** | ***E1 System can’t reach database***  1. The Error 500 screen page is shown to the user | | | |
| **Priority:** | High | | | |
| **Frequency of Use:** | High | | | |
| **Business Rules:** | - Users must provide complete and accurate information where required.  - Request details should align with the platform’s standards and guidelines. | | | |
| **Other Information:** | N/A | | | |
| **Assumptions:** |  | | | |

### 2.2 **List Mentor suggestion**

#### a. Functional Description

| **Name Of Function:** | **UC11\_List Mentor suggestion** | | | |
| --- | --- | --- | --- | --- |
| **Created By:** | AnhNHHE |  | **Created Date:** | 24/10/2024 |
| **Primary Actor:** | User(Mentee) |  | **Secondary Actor:** | System |
| **Description:** | As a user, I want to view a list of suggest mentor so that I can easily choose who i want to get support. | | | |
| **Trigger:** | User click on “suggest mentor” to redirect to page User access from url: **/happy\_programming/suggestMentor** | | | |
| **Preconditions:** | - The mentee is signed in to their account.  - The mentee access suggest mentor page | | | |
| **Postconditions:** | - A list of mentor is shown for mentee by most request | | | |
| **Normal Flow:** | 1. Mentee login and accesses the suggest mentor screen  2. Mentee view list of mentor  3. The mentee can apply filters (experience, price range) and sorting preferences (e.g., price ascending, rating descending).  4. The mentee submits the request after filling in the required information.  5. The system show the list of suggest mentor | | | |
| **Alternative Flows:** | None | | | |
| **Exceptions:** | ***E1 System can’t reach database***  1. The Error 500 screen page is shown to the user | | | |
| **Priority:** | Medium | | | |
| **Frequency of Use:** | Low | | | |
| **Business Rules:** |  | | | |
| **Other Information:** | N/A | | | |
| **Assumptions:** |  | | | |

### 2.3 Payment

#### a. Functional Description

| **Name Of Function:** | **UC12\_Create payment using VNPAY** | | | |
| --- | --- | --- | --- | --- |
| **Created By:** | AnhNHHE |  | **Created Date:** | 5/10/2024 |
| **Primary Actor:** | User(Mentee) |  | **Secondary Actor:** | System |
| **Description:** | Mentee charge money to pay mentor for their course | | | |
| **Trigger:** | User click on “Payment” button to create payment | | | |
| **Preconditions:** | - The user(mentee) is signed in their account.  - The user(mentee) access payment page | | | |
| **Postconditions:** | -Mentee charge money successfully | | | |
| **Normal Flow:** | 1. Mentee login and accesses the payment screen  2. Mentee input the amount of payment they want to charge  3. Mentee submit, then system move mentee to vnpay website to continue  4. The mentee input their banking information.  5. After success user redirect to bill | | | |
| **Alternative Flows:** | None | | | |
| **Exceptions:** | -Invalid amount | | | |
| **Priority:** | High | | | |
| **Frequency of Use:** | Medium | | | |
| **Business Rules:** |  | | | |
| **Other Information:** | N/A | | | |
| **Assumptions:** |  | | | |

### 2.4 Update request

#### a. Functional Description

| **Name Of Function:** | **UC13\_ Update request** | | | |
| --- | --- | --- | --- | --- |
| **Created By:** | AnhNHHE |  | **Created Date:** | 5/10/2024 |
| **Primary Actor:** | User(Mentee) |  | **Secondary Actor:** | System |
| **Description:** | Change request detail to get support from mentor | | | |
| **Trigger:** | User click on “Update” button in the request list of mentee | | | |
| **Preconditions:** | - The mentee is signed in their account.  - The mentee access list request page  - The mentee has created request | | | |
| **Postconditions:** | - A request is updated and saved in the database. | | | |
| **Normal Flow:** | 1. Mentee login and accesses the request list screen  2. Mentee click on request they want to update  3. Mentee enter required information and submit  4. The system validates input information  5. System update request | | | |
| **Alternative Flows:** | None | | | |
| **Exceptions:** | ***E1 System can’t reach database***  1. The Error 500 screen page is shown to the user | | | |
| **Priority:** | High | | | |
| **Frequency of Use:** | High | | | |
| **Business Rules:** | - Users must provide complete and accurate information where required.  - Request details should align with the platform’s standards and guidelines. | | | |
| **Other Information:** | N/A | | | |
| **Assumptions:** |  | | | |

### *2.5 List request by me*

#### a.Functional Description

#### 

| UC ID and Name: | **UC-14\_List request by me** | | |
| --- | --- | --- | --- |
| Created By: | HiepDD | Date Created: | 29/10/2024 |
| Primary Actor: | Mentee | Secondary Actors: | None |
| Trigger: | **Mentee clicks list request button from the navigation bar** | | |
| Description: | View list request | | |
| Preconditions: | 1. Mentee account has been created 2. Mentee must login to system 3. Request must created | | |
| Postconditions: | 1. System successfully load into list request page 2. System unsuccessfully load into list request page | | |
| Normal Flow | **List requests with actions**  1. Mentee login and accesses home screen  2. Mentee click view list request button from the navigation bar  3. System show list request page with existed information | | |
| Alternative Flows: | 1.1 Mentee login and accesses home screen  1.2 Mentee enter link to list request page | | |
| Exceptions: | 3. Data fail to load(ER01) | | |
| Priority: | High | | |
| Frequency of Use: | High | | |
| Business Rules: | None | | |
| Other Information: | None | | |
| Assumptions: | None | | |

#### b.Error message

| **ID** | **Error Definition** |
| --- | --- |
| ER01 | Data fail to load |

### 

### 2.6 Statistic request by me

#### a.Functional Description

#### 

| UC ID and Name: | **UC-15\_Statistic request by me** | | |
| --- | --- | --- | --- |
| Created By: | HiepDD | Date Created: | 29/10/2024 |
| Primary Actor: | Mentee | Secondary Actors: | None |
| Trigger: | **Mentee clicks list rate & comment button from the navigation bar** | | |
| Description: | View list request | | |
| Preconditions: | 1. Mentee account has been created 2. Mentee must login to system 3. Course must be finished at least one | | |
| Postconditions: | 1. System successfully load into list mentor page to rate 2. System unsuccessfully load into list request page | | |
| Normal Flow | **List requests with actions**  1. Mentee login and accesses home screen  2. Mentee click view list request button from the navigation bar  3. System show list request page with existed information | | |
| Alternative Flows: | 1.1 Mentee login and accesses home screen  1.2 Mentee enter link to list request page | | |
| Exceptions: | 3. Data fail to load(ER01) | | |
| Priority: | High | | |
| Frequency of Use: | High | | |
| Business Rules: | None | | |
| Other Information: | None | | |
| Assumptions: | None | | |

#### b.Error message

| **ID** | **Error Message** |
| --- | --- |
| ER01 | Data fail to load |

### 2.7 Rate & Comment

#### a.Functional Description

#### 

| ID and Name: | **UC-16\_Rate & Comment Mentor** | | |
| --- | --- | --- | --- |
| Created By: | HiepDD | Date Created: | 06/10/2024 |
| Primary Actor: | Mentee | Secondary Actors: | None |
| Description: | As a mentee, I want to be able to review mentors so that I can publish needed information for others. | | |
| Trigger: | Users click to choose mentor from mentor list then leaving the review and rate | | |
| Preconditions: | 1. Mentee are logging their account | | |
| Postconditions: | 1. Mentee left their comment & rate successfully | | |
| Normal Flow: | **Rate & Comment Mentor**   1. Mentee choose one from many mentors in list to comment 2. Give rate & comment for chosen mentor | | |
| Alternative Flows: | If mentee don’t fill anything or one of it, it will display blank need to fill and request user refill information. | | |
| Exceptions: | ***E1 System can’t reach database***  1. The Error 500 screen page is shown to the user | | |
| Priority: | High | | |
| Frequency of use | Sometimes | | |
| Business Rules: | None | | |
| ID and Name: | **UC-6 Rate & Comment Mentor** | | |
| Created By: | HiepDD | Date Created: | 06/10/2024 |

### 2.8 Update Mentee Profile

#### a.Functional Description

| ID and Name: | **UC-31\_Update Mentee Profile** | | |
| --- | --- | --- | --- |
| Created By: | HiepDD | Date Created: | 06/10/2024 |
| Primary Actor: | Mentee | Secondary Actors: | None |
| Description: | As a mentor and mentee, we want to be able to update our profile so that we can have exact information. | | |
| Trigger: | Users click to avatar button from the head page then clicking to edit profile | | |
| Preconditions: | Mentor and Mentee are logging their account | | |
| Postconditions: | Mentor, Mentee change their profile successfully | | |
| Normal Flow: | **Update Profile as User**   1. Clicking to my profile. 2. Change needed information. 3. Press Save Changes to update to database. | | |
| Alternative Flows: | If user miss some data that is empty, it will display to fill and request user fill correct information (BR01). | | |
| Exceptions: | ***E1 System can’t reach database***  1. The Error 500 screen page is shown to the user | | |
| Priority: | High | | |
| Frequency of use | Sometimes | | |
| Business Rules: | None | | |

## 3. Mentor

### 3.1 Create CV of mentor

#### a.Functional Description

#### 

| UC ID and Name: | **UC-16 Create CV of mentor** | | |
| --- | --- | --- | --- |
| Created By: | Khanhnn | Date Created: | 6/10/2024 |
| Primary Actor: | Mentor | Secondary Actors: | None |
| Trigger: | **User clicks Create CV button from the CV list page** | | |
| Description: | Create CV of Mentor | | |
| Preconditions: | 1. Mentor account has been created 2. Mentor must login to system | | |
| Postconditions: | 1. Mentor successfully create cv successfully into cv list page 2. Mentor unsuccessfully create cv and show error in cv list page | | |
| Normal Flow | **Create CV of mentor**  1. Mentor login and accesses the mentor cv list screen from navbar (BR03)  2. Mentor click on button Create CV  3. System show Create CV page with existed information  4. The system validates input information (BR01)(BR02)  5. Mentor clicks the Save change button  6. System update information into database | | |
| Alternative Flows: | 5.2 Mentor clicks the Cancel button  6.2 The system directs the user to cv list | | |
| Exceptions: | 4. Avatar oversize (ER01)  6.System create fail(ER02)(ER03) | | |
| Priority: | High | | |
| Frequency of Use: | Medium | | |
| Business Rules: | None | | |
| Other Information: | None | | |
| Assumptions: | None | | |

#### b.Business rule

| **ID** | **Rule Definition** |
| --- | --- |
| BR01 | Avatar file must smaller than 5mb |
| BR02 | Price must>1000, Year of experience must<90 |
| BR03 | Cannot delete CV if there is a course |

#### c.Error message

| **ID** | **Error Message** |
| --- | --- |
| ER01 | File size exceeds the 5 MB limit. Please upload a smaller file. |
| ER02 | 500 Database Error |
| ER03 | Unable to update your CV. Please try again. |

### 3.2 Update CV of mentor

#### a.Functional Description

#### 

| UC ID and Name: | **UC-17 Update CV of mentor** | | |
| --- | --- | --- | --- |
| Created By: | Khanhnn | Date Created: | 22/09/2024 |
| Primary Actor: | Mentor | Secondary Actors: | None |
| Trigger: | User clicks Update CV button from the page header, or  User accesses an authenticated feature (from a link or type the page URL directly into the address bar) | | |
| Description: | This screen help mentor to update created CV | | |
| Preconditions: | Mentor account has been created & authorized  CV has been created  Skills has been created | | |
| Postconditions: | Mentor update the cv successfully | | |
| Normal Flow | **Update CV of mentor**  1. Mentor login and accesses the mentor cv list screen(BR03)  2. Mentor click on button Update CV  3. System show Update CV page with existed information  4. The system validates input information (BR01,BR02)  5. Mentor clicks the Save change button  6. System update information into database | | |
| Alternative Flows: | 5.2 Mentor clicks the Cancel button  6.2 The system directs the user to cv list | | |
| Exceptions: | 4.1 Avatar oversize (ER01)  6. System update fail(ER02)(ER03) | | |
| Priority: | High | | |
| Frequency of Use: | Medium | | |
| Business Rules: | None | | |
| Other Information: | None | | |
| Assumptions: | None | | |

#### b.Business rule

| **ID** | **Rule Definition** |
| --- | --- |
| BR01 | Avatar file must smaller than 5mb |
| BR02 | Price must>1000, Year of experience must<90 |
| BR03 | Cannot delete CV if there is a course |

#### 

#### c.Error message

| **ID** | **Error Message** |
| --- | --- |
| ER01 | File size exceeds the 5 MB limit. Please upload a smaller file. |
| ER02 | 500 Database Error |
| ER03 | Unable to update your CV. Please try again. |

### 3.3 View statistic request

#### a.Functional Description

#### 

| UC ID and Name: | **UC-18 View statistic request** | | |
| --- | --- | --- | --- |
| Created By: | Khanhnn | Date Created: | 6/10/2024 |
| Primary Actor: | Mentor | Secondary Actors: | None |
| Trigger: | **Mentor clicks Statistic request button from the navigation bar** | | |
| Description: | View statistic request | | |
| Preconditions: | 1. Mentor account has been created 2. Mentor must login to system 3. Request must created 4. Mentee must exist | | |
| Postconditions: | 1. System successfully load into view statistic request page 2. System unsuccessfully load into view statistic request page and show error in cv list page | | |
| Normal Flow | **Create CV of mentor**  1. Mentor login and accesses home screen  2. Mentor click statistic request button from the navigation bar  3. System show statistic request page with existed information | | |
| Alternative Flows: | 1.1 Mentor login and accesses home screen  2.1 Mentor enter link to statistic request page  1.2 Mentor login and accesses home screen  2.2 Mentor enter any item on dashboard  3.2 Mentor click statistic request in left side bar | | |
| Exceptions: | 3. Data fail to load(ER01) | | |
| Priority: | Medium | | |
| Frequency of Use: | Medium | | |
| Business Rules: | None | | |
| Other Information: | None | | |
| Assumptions: | None | | |

#### **b.Error message**

| **ID** | **Error Definition** |
| --- | --- |
| ER01 | Data fail to load |

### 3.4 List of inviting request

#### a.Functional Description

#### 

| UC ID and Name: | **UC-19 List of inviting request** | | |
| --- | --- | --- | --- |
| Created By: | HiepDD | Date Created: | 29/10/2024 |
| Primary Actor: | Mentor | Secondary Actors: | None |
| Trigger: | **Mentor clicks inviting request button from the navigation bar** | | |
| Description: | View inviting request | | |
| Preconditions: | 1. Mentor account has been created 2. Mentor must login to system 3. Request must created 4. Mentee must exist | | |
| Postconditions: | 1. System successfully load into view inviting request page 2. System unsuccessfully load into view statistic request page | | |
| Normal Flow | **View inviting request with actions**  1. Mentor login and accesses home screen  2. Mentor click view inviting request button from the navigation bar  3. System show statistic request page with existed information | | |
| Alternative Flows: | 1.1 Mentor login and accesses home screen  1.2 Mentor enter link to view inviting request page | | |
| Exceptions: | 3. Data fail to load(ER01) | | |
| Priority: | High | | |
| Frequency of Use: | High | | |
| Business Rules: | None | | |
| Other Information: | None | | |
| Assumptions: | None | | |

#### b.Error message

| **ID** | **Error Definition** |
| --- | --- |
| ER01 | Data fail to load |

### 3.5 CRUD Slot of Mentor

#### a.Functional Description

#### 

| UC ID and Name: | **UC-30 CRUD Slot of Mentor** | | |
| --- | --- | --- | --- |
| Created By: | Khanhnn | Date Created: | 29/10/2024 |
| Primary Actor: | Mentor | Secondary Actors: | None |
| Trigger: | **Mentor clicks weekly slot button from the navigation bar** | | |
| Description: | Create, Read, Update, Delete Slot of mentor | | |
| Preconditions: | 1. Mentor account has been created 2. Mentor must login to system | | |
| Postconditions: | 1. System successfully load into CRUD Slot of Mentor 2. System unsuccessfully load into CRUD Slot of Mentor | | |
| Normal Flow | **CRUD Slot of Mentor**  1. Mentor login and accesses home screen  2. Mentor click weekly slot button from the navigation bar  3. System show CRUD Slot of Mentor page  4. Mentor clicks create slot (BR03)  5. Mentor input slot and save change  6. Mentor update slot and save change (BR01)(BR03)  7. Mentor delete slot (BR01)(BR02) | | |
| Alternative Flows: | 1.1 Mentor login and accesses home screen  1.2 Mentor click CV button from the navigation bar  1.3 Mentor click Weekly Slot button from the navigation bar  1.4 Mentor click Approved Schedule or Draft Schedule  1.5 System show CRUD Slot of Mentor page | | |
| Exceptions: | 3. Display nothing (ER500)  5. Slot already exist(ER01)  5. Cannot create new slot (ER01,ER02,ER03) | | |
| Priority: | High | | |
| Frequency of Use: | High | | |
| Other Information: | None | | |
| Assumptions: | None | | |

#### b.Business rule

| **ID** | **Rule Definition** |
| --- | --- |
| BR01 | When delete or update approved slot, system copy approved slot and save as draft slot with updated slot or deleted slot |
| BR02 | Cannot delete slot if there is a course |
| BR03 | Cannot create duplicated slot |

#### 

#### c.Error message

| **ID** | **Error Message** |
| --- | --- |
| ER01 | Unable to create the slot because a duplicate start time was found. Please choose a different time! |
| ER02 | Unable to create the slot because your CV has not been activated. Please wait for the manager to approve your CV! |
| ER03 | Unable to create your slot in the Slot draft. Please try again. |

### 3.6 Attendance Schedule

#### a.Functional Description

#### 

| UC ID and Name: | **UC-32\_Attendance Schedule** | | |
| --- | --- | --- | --- |
| Created By: | HiepDD | Date Created: | 06/11/2024 |
| Primary Actor: | Mentor | Secondary Actors: | None |
| Trigger: | **Mentor clicks Schedule button from the navigation bar** | | |
| Description: | Attendance for Mentee | | |
| Preconditions: | 1. Mentor account has been created 2. Mentor must login to system 3. Request must created 4. Mentee must exist | | |
| Postconditions: | 1. System successfully load into attendance schedule page 2. System unsuccessfully load into view statistic request page | | |
| Normal Flow | **View attendance schedule with actions**  1. Mentor login and accesses home screen  2. Mentor click schedule button from the navigation bar  3. System show attendance schedule page with registered slot information  4. Mentor marks attendance for Mentee is Attend or Absent | | |
| Alternative Flows: | 1.1 Mentor login and accesses home screen  1.2 Mentor enter link to view attendance schedule page | | |
| Exceptions: | 3. Data fail to load(ER01) | | |
| Priority: | High | | |
| Frequency of Use: | High | | |
| Business Rules: | None | | |
| Other Information: | None | | |
| Assumptions: | None | | |

#### b.Error message

| **ID** | **Error Definition** |
| --- | --- |
| ER01 | Data fail to load |

## 4. Admin

### 4.1 Admin add new Skill

#### a. Functional Description

| UC ID and Name: | **UC-20 Add new Skil** | | |
| --- | --- | --- | --- |
| Created By: | HuyMT | Date Created: | 19/09/2024 |
| Primary Actor: | Admin | Secondary Actors: | None |
| Trigger: | 1. Admin clicks the "Add Skill" button on the admin page. 2. Admin accesses the "Add Skill" feature via a direct link or URL entry. | | |
| Description: | As a admin I want to add new skill for my web so that mentor and mentee have skill they need | | |
| Preconditions: | Admin account has been created & authorized | | |
| Postconditions: | * Admin add new skill successfully | | |
| Normal Flow | **Add new skill**   1. Admin logs into the system and accesses the admin control screen. 2. Admin navigates to the "Skill Management" screen. 3. System loads and displays the list of existing skills from the database. 4. Admin clicks the "Add Skill" button on the top-right corner of the screen. 5. Admin enters skill information, including:  * Skill Name * Skill Image File * Status (default: Active) * Description  1. Admin clicks the "Add Skill" button at the bottom of the form. 2. System validates inputs, saves the new skill, and displays it on the Skill Management screen. | | |
| Alternative Flows: | None | | |
| Exceptions: | **E1. Missing Required Fields**   * If the Skill Name, Skill Image file, or Description is empty, the system displays a prompt indicating that all fields must be completed to proceed.   **E2. Duplicate Skill Name**   * If the Skill Name already exists in database, te system display notify the Skill Name already exist in Database | | |
| Priority: | High | | |
| Frequency of Use: | Medium | | |
| Business Rules: | **Authorization Rule**: Only users with an Admin role can access the add skill. If a user without this role attempts to access the add skill, they should be redirected to an unauthorized access page. | | |
| Other Information: | None | | |
| Assumptions: | None | | |

### 4.2 Admin update Skill

#### a.Functional Description

| UC ID and Name: | **UC-21 Update Skil Information** | | |
| --- | --- | --- | --- |
| Created By: | HuyMT | Date Created: | 19/09/2024 |
| Primary Actor: | Admin | Secondary Actors: | None |
| Trigger: | 1. Admin clicks the "Update" button on the admin page. 2. Admin accesses the "Update Skill" feature via a direct link or URL entry. | | |
| Description: | As an admin I want to update skill information for my web so that skill can have new data needed | | |
| Preconditions: | * Admin account has been created & authorized * The skill must already exist in database | | |
| Postconditions: | * Skill information is updated successfully in the system. | | |
| Normal Flow | **Update skill information**   1. Admin logs into the system and accesses the admin control screen. 2. Admin navigates to the "Skill Management" screen. 3. System loads and displays the list of skills from the database. 4. Admin clicks "Update" on the specific skill they want to modify. 5. System loads the current data for that skill, including:  * Skill ID * Skill Name * Status * Image * Description  1. Admin edits the desired fields, including:  * Skill Name * Skill Image file * Status * Description  1. Admin clicks the "Update Skill" button at the bottom of the form. 2. System validates inputs, updates the skill information in the database, and displays the updated information on the Skill Management screen | | |
| Alternative Flows: | None | | |
| Exceptions: | * **E1. Missing Required Fields**   + If the Skill Name, Skill Image Link, or Description field is left empty, the system prompts the admin to complete these fields before submission. * **E2. Database Connection Failure**   + If the system cannot connect to the database, it displays a "500 Internal Server Error" page to the admin. | | |
| Priority: | High | | |
| Frequency of Use: | Medium | | |
| Business Rules: | **Authorization Rule**: Only users with an Admin role can access the update skill. If a user without this role attempts to access the update skill, they should be redirected to an unauthorized access page. | | |
| Other Information: | None | | |
| Assumptions: | None | | |

#### b.Business Rules

### 4.3 Admin view list of skill

#### a.Functional Description

| UC ID and Name: | **UC-22 Admin view list of skill** | | |
| --- | --- | --- | --- |
| Created By: | HuyMT | Date Created: | 30/09/2024 |
| Primary Actor: | Admin | Secondary Actors: | None |
| Trigger: | 1. Admin clicks "Skill List" on the admin page. 2. Admin accesses the "Skill List" feature via a direct link or URL entry. | | |
| Description: | As an admin I want to see list of skill of my web so that i can manage skill | | |
| Preconditions: | Admin account is created, logged in, and authorized. | | |
| Postconditions: | * The skill list is successfully loaded and displayed. | | |
| Normal Flow | **View skill list information**   1. Admin logs into the system and accesses the admin control screen. 2. Admin navigates to the "Skill Management" screen. 3. System retrieves and displays the list of skills from the database. 4. System displays current data for each skill, including:  * Skill ID * Skill Name * Status * Image Link and Image * Description | | |
| Alternative Flows: | **A1. Search for a Skill**   * If the admin wants to search for a specific skill, they can enter the skill name in the search bar. * System filters and displays skills matching the search criteria.   **A2. Adjust Displayed Skills per Page**   * Admin can choose the number of skills to display on a single page. * System updates the view based on the selected display setting. | | |
| Exceptions: | **E1. Database Connection Failure**   * If the system cannot connect to the database, it displays a "500 Internal Server Error" page to the admin. | | |
| Priority: | High | | |
| Frequency of Use: | Medium | | |
| Business Rules: | **Authorization Rule**: Only users with an Admin role can access the skill list. If a user without this role attempts to access the skill list, they should be redirected to an unauthorized access page. | | |
| Other Information: | None | | |
| Assumptions: | None | | |

#### b.*Business Rules*

### 4.4 View list of mentor

#### a. Functional Description

| UC ID and Name: | **UC-23 View list of Mentor** | | |
| --- | --- | --- | --- |
| Created By: | HuyMT | Date Created: | 3/10/2024 |
| Primary Actor: | Admin | Secondary Actors: | None |
| Trigger: | 1. Admin clicks "Extend User List" and selects "Mentor." 2. Admin accesses the mentor list via a direct link or URL entry. | | |
| Description: | As a admin I want to see list of Mentor so that i can manage Mentor | | |
| Preconditions: | Admin account has been created & authorized | | |
| Postconditions: | * The mentor list is successfully loaded and displayed. | | |
| Normal Flow | **View mentor information**   1. Admin logs into the system and accesses the admin control screen. 2. Admin navigates to the "Mentor List" screen. 3. System retrieves and displays the list of mentors from the database. 4. System displays current data for each mentor, including:  * Mentor ID * Mentor Name * Profession * Current accepted requests * Rating | | |
| Alternative Flows: | **A1. Search for a Mentor**   * If the admin wants to search for a specific mentor, they can search by mentor ID, name, or profession. * System filters and displays mentors matching the search criteria.   **A2. Adjust Displayed Mentors per Page**   * Admin can select the number of mentors to display on a single page. * System updates the view based on the selected display setting. | | |
| Exceptions: | **E1. Database Connection Failure**   * If the system cannot connect to the database, it displays a "500 Internal Server Error" page to the admin. | | |
| Priority: | High | | |
| Frequency of Use: | Medium | | |
| Business Rules: | **Authorization Rule**: Only users with an Admin role can access the mentor list. If a user without this role attempts to access the mentor list, they should be redirected to an unauthorized access page. | | |
| Other Information: | None | | |
| Assumptions: | None | | |

#### b.Business Rules

### 4.5 View list of mentee

#### a.Functional Description

| UC ID and Name: | **UC-24 View list of Mentee** | | |
| --- | --- | --- | --- |
| Created By: | HuyMT | Date Created: | 10/10/2024 |
| Primary Actor: | Admin | Secondary Actors: | None |
| Trigger: | * Admin clicks "Extend User List" and selects "Mentee." * Admin accesses the mentee list via a direct link or URL entry. | | |
| Description: | As an admin, I want to view a list of mentees so that I can manage their information effectively. | | |
| Preconditions: | Admin account is created, logged in, and authorized. | | |
| Postconditions: | * The mentee list is successfully loaded and displayed. | | |
| Normal Flow | **View mentee list information**  Admin logs into the system and accesses the admin control screen.  Admin navigates to the "Mentee List" screen.  System retrieves and displays the list of mentees from the database.  System displays relevant data for each mentee, including:   * Mentee ID * Mentee Name * Number of mentor * Total hour * Total Skill | | |
| Alternative Flows: | **A1. Access via Direct URL**   * Admin can directly enter the URL to access the mentee list page. * System loads and displays the mentee list if the admin is authenticated. | | |
| Exceptions: | **E1. Database Connection Failure**   * If the system cannot connect to the database, the mentee list fails to load. * The system displays an error message or a "500 Internal Server Error" page to the admin. | | |
| Priority: | High | | |
| Frequency of Use: | Medium | | |
| Business Rules: | **Authorization Rule**: Only users with an Admin role can access the mentee list. If a user without this role attempts to access the mentee list, they should be redirected to an unauthorized access page. | | |
| Other Information: | None | | |
| Assumptions: | None | | |

#### b.Business Rules

### 4.6 View all request of all mentee

#### a. Functional Description

| UC ID and Name: | **UC-25 View all request of all Mentee** | | |
| --- | --- | --- | --- |
| Created By: | HuyMT | Date Created: | 15/10/2024 |
| Primary Actor: | Admin | Secondary Actors: | None |
| Trigger: | * Admin clicks "Request List" in the admin panel. * Admin accesses the request list directly via a URL entry. | | |
| Description: | As an admin, I want to view all requests from all mentees so that I can manage their requests effectively. | | |
| Preconditions: | Admin account is created, logged in, and authorized. | | |
| Postconditions: | * The request list is successfully loaded and displayed. | | |
| Normal Flow | **View request list information**   1. Admin logs into the system and accesses the admin control screen. 2. Admin navigates to the "Request List" screen. 3. System retrieves and displays all requests from the database. 4. System shows detailed information for each request, including:  * Request ID * Mentee ID * Request title (if applicable) * Status | | |
| Alternative Flows: | **A1. Search for Requests**   * Admin can enter search criteria, such as mentee name, request title, creation date, or request ID, to filter the list of requests. * System filters and displays only the requests that match the search criteria. | | |
| Exceptions: | **E1. Database Connection Failure**   * If the system cannot connect to the database, it fails to load the request list. * The system displays an error message or a "500 Internal Server Error" page to the admin. | | |
| Priority: | High | | |
| Frequency of Use: | Medium | | |
| Business Rules: | **Authorization Rule**: Only users with an Admin role can access the request list. If a user without this role attempts to access the request list, they should be redirected to an unauthorized access page. | | |
| Other Information: | None | | |
| Assumptions: |  | | |

### 

### 4.7 Admin dashboard

#### a. Functional Description

| UC ID and Name: | **UC-26 Admin Dashboard** | | |
| --- | --- | --- | --- |
| Created By: | HiepDD | Date Created: | 05/11/2024 |
| Primary Actor: | Admin | Secondary Actors: | None |
| Trigger: | * Admin clicks "Dashboard" in the admin panel. * Admin accesses the admin dashboard directly via a URL entry. | | |
| Description: | As an admin, I want to view all statistics and actions from all users so that I can manage their accounts effectively. | | |
| Preconditions: | Admin account is created, logged in, and authorized. | | |
| Postconditions: | * The statistic and user list is successfully loaded and displayed. | | |
| Normal Flow | **View request list information**   1. Admin logs into the system and accesses the admin control screen. 2. Admin navigates to the "Dashboard" screen. 3. System retrieves and displays all users from the database. 4. System shows detailed information for all users, including:  * Total Mentor * Total Mentee * Newest registered users * Mentor, Mentee account | | |
| Alternative Flows: | **A1. Active or Inactive**   * Admin can activate or deactivate Mentor or Mentee accounts. * System does the action and displays Mentor or Mentee by status. | | |
| Exceptions: | **E1. Database Connection Failure**   * If the system cannot connect to the database, it fails to load the user list. * The system displays an error message or a "500 Internal Server Error" page to the admin. | | |
| Priority: | Medium | | |
| Frequency of Use: | Medium | | |
| Business Rules: | **Authorization Rule**: Only users with an Admin role can access the request list. If a user without this role attempts to access the dashboard, they should be redirected to an unauthorized access page. | | |
| Other Information: | None | | |
| Assumptions: |  | | |

### 

## 5. Manager

### 5.1 Manager CV & Slot of mentor

#### a.Functional Description

#### 

| UC ID and Name: | **UC-26\_Manager CV & Slot of mentor** | | |
| --- | --- | --- | --- |
| Created By: | Khanhnn | Date Created: | 6/10/2024 |
| Primary Actor: | Manager | Secondary Actors: | None |
| Trigger: | **Manager clicks mentor management from manager navbar** | | |
| Description: | Manager CV & Slot of mentor | | |
| Preconditions: | 1. Manager account has been created 2. Manager must login to system | | |
| Postconditions: | 1. System successfully display cv list of mentor 2. System unsuccessfully display cv list of mentor and show error | | |
| Normal Flow | **Create CV of mentor**  1. Manager login and accesses the mentor cv list screen  2. Manager click on button manager cv  3. System show manager cv page with existed information  4. Manager clicks view to see popup cv detail  5. Manager clicks the Active/Inactive cv button (BR01,BR02)  6. Manager clicks the Schedule Icon  7. Manager clicks View to see popup slot detail  8. Manager click (tick) icon to active or (x) icon inactive slot (BR02,BR03) | | |
| Alternative Flows: | 1.1 Manager insert link to manager cv page  2.1 The system directs the user to cv list | | |
| Exceptions: | 5. Active cv of mentor have exist active cv (ER01)  8. Active or Inactive Slot have active request (ER02),(ER03) | | |
| Priority: | High | | |
| Frequency of Use: | Medium | | |
| Business Rules: | Only one cv can be activated  Slot have active request cant change status | | |
| Other Information: | None | | |
| Assumptions: | None | | |

#### **b.Business rule**

| **ID** | **Rule Definition** |
| --- | --- |
| BR01 | Mentor have only one activated CV |
| BR02 | Cannot deactivate CV or Slot when Mentor have studying courses |
| BR03 | When deactivate slot mean delete all draft slot |

#### **c.Error message**

| **ID** | **Error Definition** |
| --- | --- |
| ER01 | Mentor already have activated CV |
| ER02 | Mentor already have activated Request, cannot activate new Slot |
| ER03 | Mentor already have activated Request, cannot deactivate Slot |

### 5.2 Manger payment request

#### a.Functional Description

#### 

| UC ID and Name: | **UC-27 Manger payment request** | | |
| --- | --- | --- | --- |
| Created By: | Khanhnn | Date Created: | 6/10/2024 |
| Primary Actor: | Manager | Secondary Actors: | None |
| Trigger: | **Manager clicks payment request button from manager navbar** | | |
| Description: | Manger payment request | | |
| Preconditions: | 1. Manager account has been created 2. Manager must login to system | | |
| Postconditions: | 1. System successfully display request list 2. System unsuccessfully display request list and show error | | |
| Normal Flow | **Create CV of mentor**  1. Manager login and accesses the manger payment request screen  2. Manager click on button payment request  3. System show manger payment request page with existed information  4. Manager clicks Detail to see payment detail  5. Manager clicks the (tick) icon next to Pending/Studying status to change status request (BR01)  6. Manager clicks the Detail button to see status detail of completed request  7. Manager click Make Payment button to pay mentor (BR02)(BR03) | | |
| Alternative Flows: | 1.1 Manager insert link to manager payment request  2.1 The system directs the user to Manager payment request page | | |
| Exceptions: | 1. Can't change status(ER01)(ER500) 2. Can’t make payment(ER01) | | |
| Priority: | High | | |
| Frequency of Use: | Medium | | |
| Business Rules: | None | | |
| Other Information: | None | | |
| Assumptions: | None | | |

#### **b.Business rule**

| **ID** | **Rule Definition** |
| --- | --- |
| BR01 | Only can change status processing to studying |
| BR02 | Manager will keep 20% of request total money |
| BR03 | When manager pay mentor, status change to paid |

#### **c.Error message**

| **ID** | **Error Message** |
| --- | --- |
| ER01 | Failed to change status. |
| ER02 | Cannot change status from 'Studying' to 'Processing'. |

## 6. Common

### 6.1 View Slot

#### a.Functional Description

#### 

| UC ID and Name: | **UC-28 View Slot of Mentee** | | |
| --- | --- | --- | --- |
| Created By: | Khanhnn | Date Created: | 6/10/2024 |
| Primary Actor: | Mentee | Secondary Actors: | Manager |
| Trigger: | **User clicks slot button from navbar** | | |
| Description: | View Slot of mentor or mentee | | |
| Preconditions: | 1. Mentor and Mentee account has been created 2. User must login to system 3. Request status Studying must exist 4. Slot must exist | | |
| Postconditions: | 1. System successfully display slot 2. System unsuccessfully display slot and show error | | |
| Normal Flow | **Create CV of mentor**  1. User login and accesses the View Slot screen  2. User click on nav button to change month/week/day  3. System show slot with existed information  4. User click number on calendar to see slot detail | | |
| Alternative Flows: | 1.1 User insert link to view slot page  2.1 The system directs the view slot | | |
| Exceptions: | None | | |
| Priority: | Medium | | |
| Frequency of Use: | Medium | | |
| Business Rules: | None | | |
| Other Information: | None | | |
| Assumptions: | None | | |

### 6.2 **Authorization**

#### a.Functional Description

#### 

| UC ID and Name: | **UC-29 Authorization** | | |
| --- | --- | --- | --- |
| Created By: | HuyMT | Date Created: | 27/10/2024 |
| Primary Actor: | System | Secondary Actors: | None |
| Trigger: | **User access any page** | | |
| Description: | Authorization make only user with specific row can access specific page | | |
| Preconditions: | 1. Login user have been authorized | | |
| Postconditions: | 1. System redirect user to specific page base on user’s role | | |
| Normal Flow | **System authorization**  1. User login and accesses specific page authorized  2. System check user role and if user accessible that page  3. System redirect user to that page if they have permission | | |
| Alternative Flows: | 1.1 User insert direct URL link  2.1 The system check user authorized and redirect them | | |
| Exceptions: | None | | |
| Priority: | Medium | | |
| Frequency of Use: | Medium | | |
| Business Rules: | None | | |
| Other Information: | None | | |
| Assumptions: | None | | |

### 6.3 Reset Password

#### a.Functional Description

#### 

| ID and Name: | **UC-3\_Reset Password** | | |
| --- | --- | --- | --- |
| Created By: | HiepDD | Date Created: | 22/09/2024 |
| Primary Actor: | Mentor, Mentee | Secondary Actors: | None |
| Description: | As a mentor and mentee, we want to be able to reset our password when we forgot so that we can use new password to login. | | |
| Trigger: | User clicks Forgot Password button from the login page | | |
| Preconditions: | Mentor and Mentee have already had registered account | | |
| Postconditions: | Mentor, Mentee update their new password successfully | | |
| Normal Flow: | **Reset Password**   1. Mentor and Mentee click to forgot password. 2. Screen move to request reset password page. 3. Mentor and Mentee fill their registered email and username in blank. 4. Tapping to request reset button and message pop up below button. 5. Mentor and Mentee check their email account with reset random password . 6. Logging username with reset password recently then change reset password to make my own password. 7. System update their new password to database | | |
| Alternative Flows: | If user fill wrong type of email or username, webpage will display error and request user refill correct type of email and username. | | |
| Exceptions: | ***E1 System can’t reach database***  1. The Error 500 screen page is shown to the user | | |
| Priority: | High | | |
| Frequency of use | Sometimes | | |
| Business Rules: | None | | |

### 6.4 Change Password

#### a.Functional Description

#### 

| ID and Name: | **UC-4\_Change Password** | | |
| --- | --- | --- | --- |
| Created By: | HiepDD | Date Created: | 22/09/2024 |
| Primary Actor: | Mentor, Mentee | Secondary Actors: | None |
| Description: | As a mentor and mentee, we want to be able to change our password so that we can use a new password for more security. | | |
| Trigger: | Users click to avatar button from the head page then clicking to change password | | |
| Preconditions: | Mentor and Mentee are logging their account | | |
| Postconditions: | Mentor, Mentee change their new password successfully | | |
| Normal Flow: | **Change Password**   1. Mentor and Mentee click to change password at head menu. 2. Screen move to change password page. 3. Mentor and Mentee fill their username, old password, new password, confirm password in blank. 4. Tapping to Enter button to change. | | |
| Alternative Flows: | If user fill wrong username, old password, new password and confirm password will display error and request user refill correct information. | | |
| Exceptions: | ***E1 System can’t reach database***  1. The Error 500 screen page is shown to the user | | |
| Priority: | High | | |
| Frequency of use | Sometimes | | |
| Business Rules: | None | | |

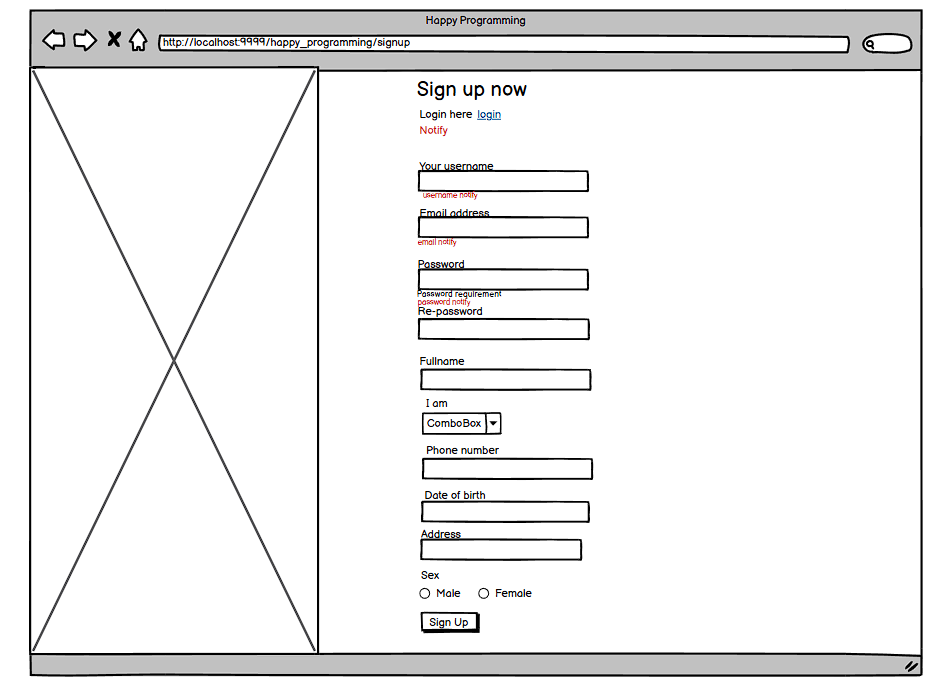
# III. Design Specifications

#### a. User Signup

This screen allows user to be authenticated to the system screens/functionalities.

Related use cases:

##### UI Design



| **Field Name** | **Field Type** | **Description** |
| --- | --- | --- |
| Your username | Text Box | This is for user to input their username |
| Email address | Mail Box | This is for user to input valid email format address for signing up |
| Address | Text Box | This is for user to input address for signing up |
| username notify | <small> | show for user if username already exist |
| password notify | <small> | notify user if password not meet requirement |
| password requirement | <small> | show password requirement |
| re-password notify | <small> | notify user if password do not match re-pass |
| email notify | <small> | notify user if email exist |
| Password | Password Box | This is for user to input password for signing up |
| Re-password | Password Box | This is for user to input re-password |
| notify | <small> | Show sign up notification |
| Login | <a href> | Link to login page |
| Fullname | Text Box | This is for user to input user’s full name |
| I am | Select Box | This is for user to input their role |
| Phone number | Number box | This is for user to input their phone number |
| Date of birth | Date time box | This is for user to input their date of birth |
| Sex | Radio box | This is for user to input their sex |
| Sign up | Button | User clicks to authenticate him/herself into the system with provided email & password |

##### Database Access

| **Table** | **CRUD** | **Description** |
| --- | --- | --- |
| User | C | Create new user |
| User | R | Check if user with email, username already exist |
| User | U | Update user by user name |

##### SQL Commands:

1/ create new user

INSERT INTO [dbo].[User]

([RoleID], [Username], [Status], [CreateDate], [Email],

[Password], [Phone], [Address], [DateOfBirth], [FullName], [Gender])

2/ Check if user with username is already exist

SELECT [RoleID], [Username], [Status], [CreateDate],

[Email], [Password], [Phone], [Address], [DateOfBirth], [FullName], [Gender]

FROM [dbo].[User] WHERE Username = ?

3/Check if user with email is already exist

SELECT [RoleID], [Username], [Status], [CreateDate],

[Email], [Password], [Phone], [Address], [DateOfBirth], [FullName], [Gender]

FROM [dbo].[User] WHERE Email = ?

4/update user

UPDATE [dbo].[User] SET

[RoleID] = ?, [Status] = ?, [CreateDate] = ?, [Email] = ?

[Password] = ?, [Phone] = ?, [Address] = ?, [DateOfBirth] = ?, [FullName] = ?, [Gender] = ?

WHERE [Username] = ?

5/Find user by username

SELECT [RoleID], [Username], [Status], [CreateDate],

[Email], [Password] "

"FROM [dbo].[User] WHERE Username = ?

6/Insert mentor

INSERT INTO Mentor (RoleID, Username, CreateDate, Phone, Address, DateOfBirth, FullName, Gender, Status)

VALUES (?, ?, GETDATE(), ?, ?, ?, ?, ?, ?)

7/Insert mentee

INSERT INTO Mentee (RoleID, Avatar, Username, CreateDate, Email, Phone, Address,

DateOfBirth, FullName, Gender, Status)

VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)

8/find user by email

SELECT [RoleID], [Username], [Status], [CreateDate],

"[Email], [Password] "

"FROM [dbo].[User] WHERE Email = ?

9/create new wallet

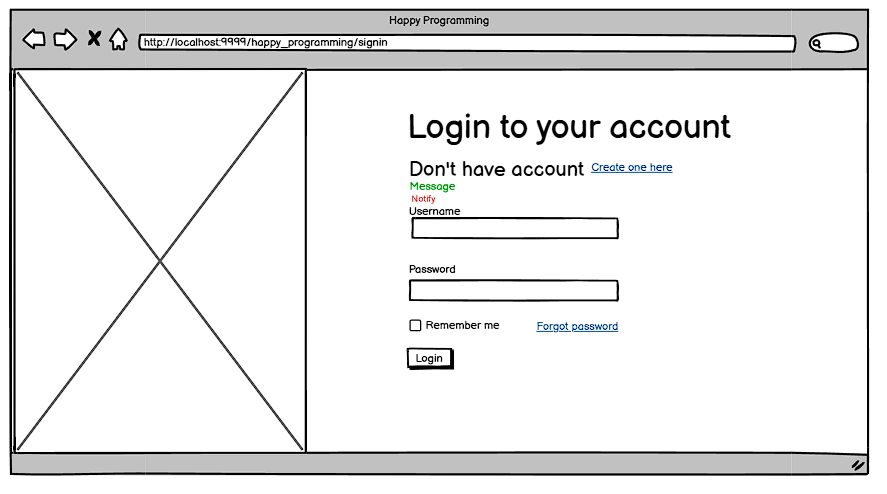
INSERT INTO [dbo].[Wallet] ([Balance],[Username],[Hold]) VALUES (0,?,0)

#### b.User sign in

This screen allows user to be authenticated to the system screens/functionalities.

Related use cases:

##### UI Design



| Username | Text Box | This is for user to input username |
| --- | --- | --- |
| Password | Password Box | This is for user to input password for signing in |
| Remember me | Check Box | This allow user to save their username and password |
| Message | <p> | Show message(if have) from system |
| Notify | <small> | Show notify (if have) from system |
| Create one here | <a href> | Link to sign up site |
| Forgot password | <a href> | Link to forgot password site |
| Login | Button | This allow user submit sign in information |

##### SQL Commands:

##### 

| **Table** | **CRUD** | **Description** |
| --- | --- | --- |
| User | R | Check if UserName and Pass correct |

1/ Check if user exist

SELECT \*

FROM [dbo].[User] WHERE Username = '' and Password = ''

2/Get mentor by username

select \* from Mentor where Username = ?

3/Get cv by mentorId

select\*from [dbo].[CV] where MentorID =? and Status='active'

4/get wallet by user name

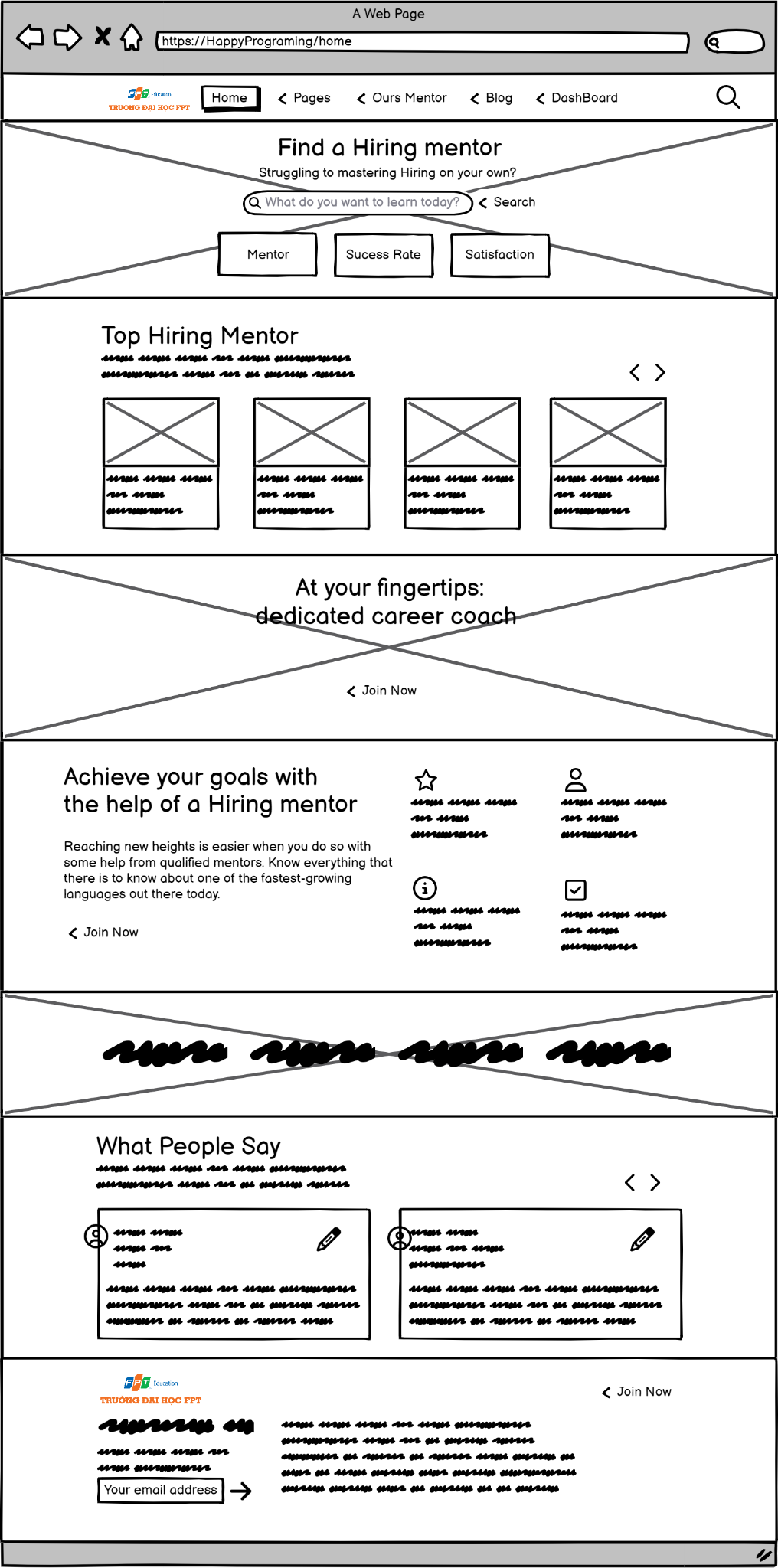
SELECT \* FROM Wallet WHERE Username = ?

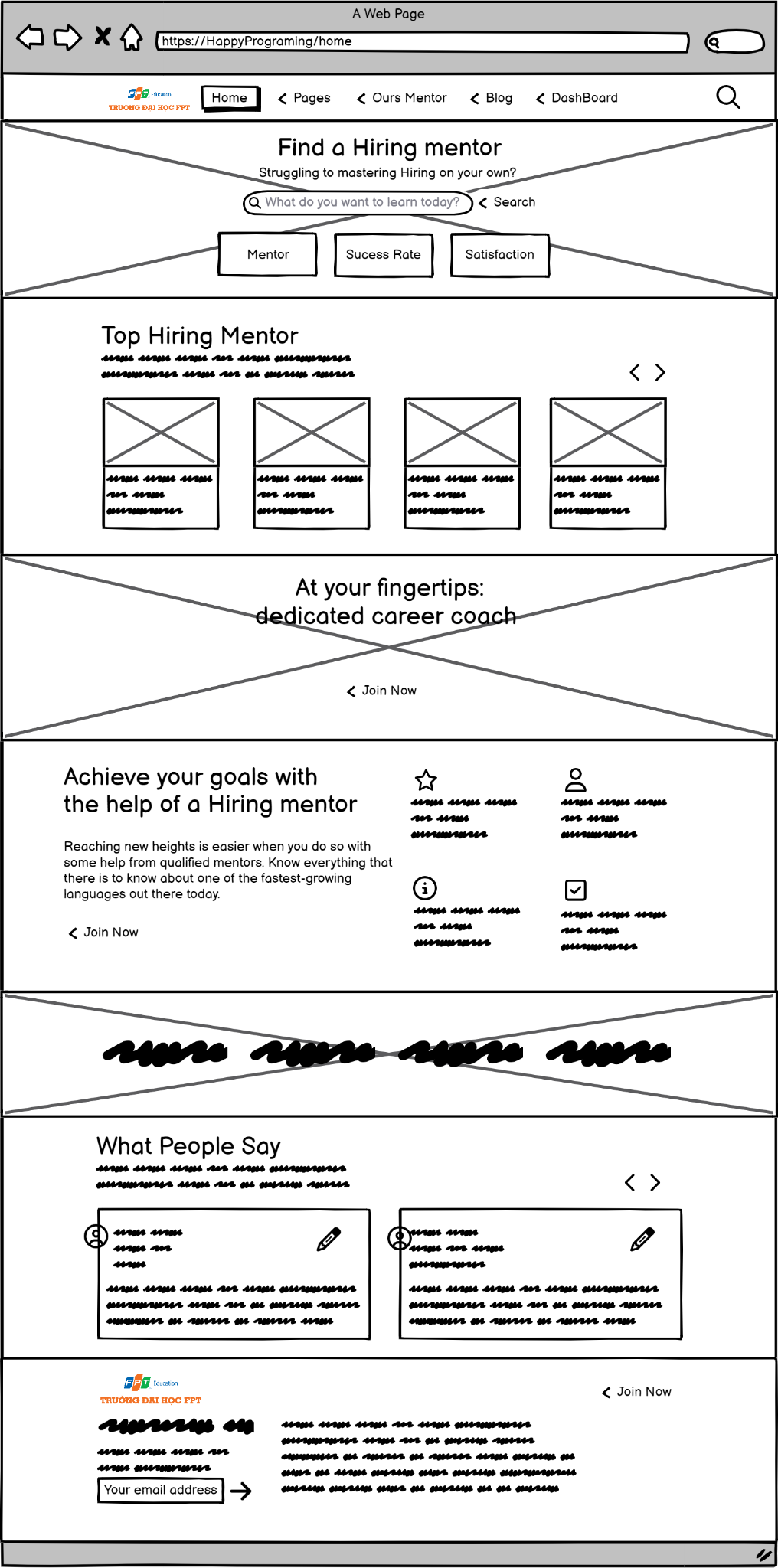
5/Find mentee by username

select \* from Mentee where Username = ?

#### c. Home Page

##### UI Design





| **Field Name** | **Field Type** | **Description** |
| --- | --- | --- |
| ***Search Fields*** | | |
| Home | Button | User click to go to home page |
| Pages | Button-  dropdown | User navigate to another pages |
| Ours Course | Button-  dropdown | User enter List of Skills |
| Blog | Button-  dropdown | User enter Blogs and Evens page |
| Dashboard | Button-  dropdown | User enter dashboard, profile, etc. |
| SearchBar | Text-box | User input search skill information |
| Join now | Button | User enter List of Skills |

##### Database Access

| **Table** | **CRUD** | **Description** |
| --- | --- | --- |
| User | R | Count number of Users, Search |
| Rate | R | statistic satisfaction |
| Skill | R | Search by Skill |
| CV | R | View popular mentors |

##### SQL command:

1. **Get top 4 list of skill by rate :**

select top 4 s.SkillID,s.SkillName,s.CreateDate,s.Description,s.Img,Avg(Rating)[Rating]

from Skill s join SkillList sl on s.SkillID=sl.SkillID

where s.Status='active'

group by s.SkillID,s.SkillName,s.CreateDate,s.Img,sl.Rating,s.Description

order by sl.Rating desc;

1. **Get list of rate**

select \* from Rate

1. **Get list of mentor**

select\*from Mentor

1. **Get list of mentee**

select\*from Mentee

1. **Count user**

SELECT COUNT(\*) FROM [dbo].[User]

1. **Count mentor**

SELECT COUNT(\*) FROM [dbo].[User] where [dbo].[User].RoleID=1

1. **Count mentee**

SELECT COUNT(\*) FROM [dbo].[User] where [dbo].[User].RoleID=2

1. **Count request**

select count(RequestID) from Request

1. **Count skills**

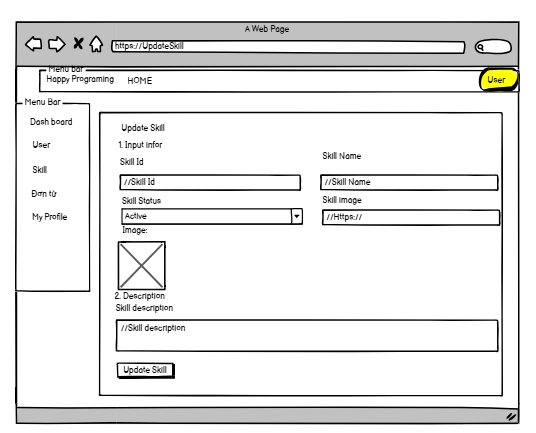
select count(SkillID) from Skill

1. **Get average rate of mentee**

SELECT AVG(CAST(Rate AS FLOAT)) FROM Rate;

#### d. Admin update Skill

##### UI Design

This show screen of update skill

| **Field Name** | **Field Type** | **Description** |
| --- | --- | --- |
| ***Search Fields*** | | |
| Skill Id | Text-box | This for admin to see Skill Id only, cannot change |
| Skill name | Text-box | This is for the admin to input skill name |
| Skill Status | Combo-box | This is for the admin to chose Status which is Active or Inactive |
| Description | Text box | This is for admin to input the description of the skill |
| Image | Text\_box | This is for admin to input their link image to illustrate the skill |
| Image | img | This field show current image of skill |
| Update Skill | Button | Admin click to submit update |
| Dash board | Link | Admin click to go to dash board page |
| Skill | Link | Admin click to go to skill list page |
| Đơn từ | Link | Admin click to go to đơn từ page |
| User | Link | Admin click to go to User list page |
| My profile | Link | Admin click to go to profile page |
| Home | Link | Admin click to go to home page |

##### Database Access

| **Table** | **CRUD** | **Description** |
| --- | --- | --- |
| Skill | R | Search skill by specific condition |
| Skill | U | Update Skill |

##### SQL Commands:

*1/ Search Skill   
SELECT [SkillID]*

*,[SkillName]*

*,[CreateDate]*

*,[Description]*

*,[Status]*

*,[Img]*

*FROM [dbo].[Skill]*

WHERE <Search Conditions,,>

2/ Update Skill Information

UPDATE [dbo].[Skill]

SET [SkillName] = <SkillName, nvarchar(50),>

,[CreateDate] = <CreateDate, date,>

,[Description] = <Description, nvarchar(max),>

,[Status] = <Status, nvarchar(20),>

,[Img] = <Img, nvarchar(max),>

WHERE <Search Conditions,,>

#### e. Admin add new skill

##### UI Design

This show screen of adding new skill



| **Field Name** | **Field Type** | **Description** |
| --- | --- | --- |
| ***Search Fields*** | | |
| Skill name | Text-box | This is for the admin to input skill name |
| Skill Status | radio | This is for the admin to input Status for skill. Which default is Active |
| Description | textarea | This is for admin to input the description of the skill |
| Image | Text-box | This is for admin to input their link image to illustrate the skill |
| Add Skill | Button | Admin click to add new skill to database |
| Dash board | Link | Admin click to go to dash board page |
| Skill | Link | Admin click to go to skill list page |
| Đơn từ | Link | Admin click to go to đơn từ page |
| User | Link | Admin click to go to User list page |
| My profile | Link | Admin click to go to profile page |
| Home | Link | Admin click to go to home page |

##### Database Access

| **Table** | **CRUD** | **Description** |
| --- | --- | --- |
| Skill | R | Search skill by name |
| Skill | U | Create new skill |

##### SQL Commands:

*1/Search Skill by name  
SELECT [SkillID]*

*,[SkillName]*

*,[CreateDate]*

*,[Description]*

*,[Status]*

*,[Img]*

*FROM [dbo].[Skill] where SkillName = ?*

2/ create new Skill

INSERT INTO [dbo].[Skill]

([SkillName]

,[CreateDate]

,[Description]

,[Status]

,[img])

VALUES

(<SkillName, nvarchar(50),>

,<CreateDate, date,>

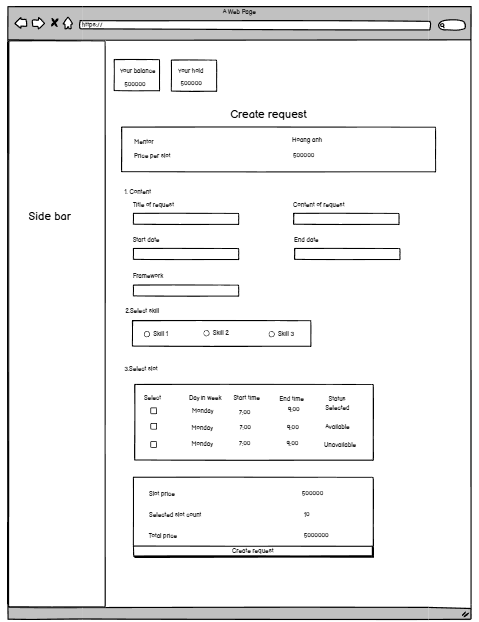
,<Description, nvarchar(max),>

,<Status, nvarchar(20),>

,<img, nvarchar(max),>)

#### g. Create request

##### UI Design



| **Field Name** | **Field Type** | **Description** |
| --- | --- | --- |
| Tittle of request | Text Box | Input tittle of request from mentee to mentor |
| Content of request | Text box | Input content of request from mentee to mentor |
| Framework | Text Box | Input framework that mentee want to learn |
| Start Date | Timebox | Input start date that mentee want to support |
| End date | Date box | Input end date that mentee want to support |
| Skill | Checkbox | Allow user to select skill |
| Select | Checkbox | Allow user to select slots |
| Create request | Submit button | Submit all information that mentee input |

##### Database Access

##### 

| **Table** | **CRUD** | **Description** |
| --- | --- | --- |
| Mentor | R | Get mentor that mentee want to create request |
| CV | R | Get CV of mentor that mentee want to create request |
| RequestSlotItem | C, R | -Get slot of mentee have status “Processing” -Add slot that mentee want to create request by requestID |
| RequestItem | C | -Add skill that mentee want to learn by requestID |
| Request | C, R | -Create new request from mentee to mentor  -Get newest request |
| Slot | R | Get list slot of mentor |
| SkillList | R | Get list skill of mentor |
| Mentee | R | Get id of mentee create request |

##### SQL Command.

1/Find mentor by id

SELECT [MentorID] ,[RoleID] ,[Username] ,[CreateDate] ,[Phone] ,[Address] ,[DateOfBirth],[FullName] ,[Gender] ,[Status]

FROM [dbo].[Mentor]

where MentorID = ?

2/Find cv by mentor id

select\*from [dbo].[CV] where MentorID =? and Status='active';

3/Get slot of mentee have status “Processing”

select \*

from RequestSlotItem rq

join Request r on rq.RequestID = r.RequestID

where r.Status = 'Processing' and r.MenteeID = ?;

4/Get list slot of mentor by mentor ID

SELECT SlotID, MentorID, StartTime, EndTime, DayInWeek, Status FROM Slot WHERE MentorID = ?

ORDER BY CASE DayInWeek

WHEN 'Monday' THEN 1

WHEN 'Tuesday' THEN 2

WHEN 'Wednesday' THEN 3

WHEN 'Thursday' THEN 4

WHEN 'Friday' THEN 5

WHEN 'Saturday' THEN 6

WHEN 'Sunday' THEN 7

ELSE 8 END";

5/Get list skill of mentor

select s.SkillID,s.SkillName from Skill s join SkillList sl on s.SkillID=sl.SkillID where MentorID=?

6/Insert new request:

INSERT INTO Request (MentorID, MenteeID, Price, Note, CreateDate, Status, Title, DeadlineHour, DeadlineDate, Framework)

VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?)

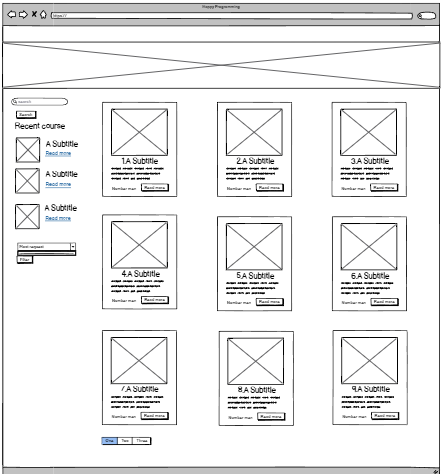
SELECT TOP 1 RequestID FROM [dbo].[Request] ORDER BY RequestID DESC

INSERT INTO [dbo].[RequestItem] (RequestID, SkillID) VALUES (?, ?)

INSERT INTO [dbo].[RequestSlotItem] (RequestID, SlotID) VALUES (?, ?)

#### h. View SkillList

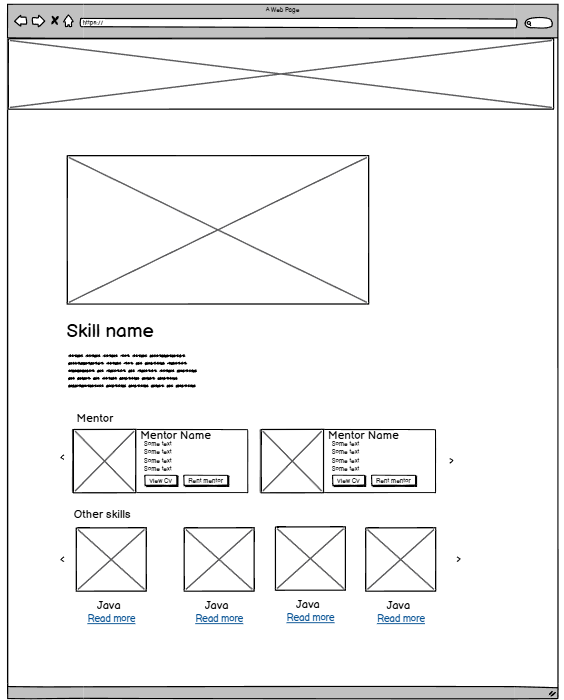
##### UI Design



| **Field Name** | **Field Type** | **Description** |
| --- | --- | --- |
| Read more | Button | Click to show more information about that skill |
| One, two, three | Button | Click to change skill page |
| Search box | Form | Search skill when user enter skill name |
| Number mentor | <p> | Show number mentor |
| … | <p> | Show skill description |
| Read more | a<href> | Click to show that skill detail |
| A subtitle | a<href> | Click to show that skill detail |
| Combo box | Select box | Select filter option |
| Filter | button | Submit filter option |
| Image | a<href><img></a> | Click to show that skill detail |
| Search | Button | Submit text to search |

#### 

Skill detail



| **Field Name** | **Field Type** | **Description** |
| --- | --- | --- |
| View cv | Button | Click to show mentor cv |
| Rent mentor | Button | Click to create request to get support from mentor |
| image | <img> | Show the image of mentor/skill |
| Skill name | <h2> | Show skill name |
| Some text | <p> | Show mentor information(cv) |
| Mentor name | <h6> | Show mentor name |
| … | <p> | Show skill description |

#### 

##### Database access:

| **Table** | **CRUD** | **Description** |
| --- | --- | --- |
| Mentor | R | Get mentor by skillid |
| CV | R | Get CV of mentor by mentor ID |
| Skill | R | Get list skill of skill, search skill |

##### SQL Commands:

1/Select ? skills in database

SELECT [SkillID] ,[SkillName], [CreateDate] ,[Description], [Status], [Img]

FROM [dbo].[Skill]

order by Skill.SkillName asc

offset ? rows

fetch next ? row only

2/Get all mentor by skil id

select \* from mentor m

join SkillList s on m.MentorID = s.MentorID

where SkillID = ?

3/Get top 3 newest skill by date

SELECT Top 3 [SkillID],

[SkillName],

[CreateDate],

[Description],

[Status],

[Img]

FROM [dbo].[Skill]

ORDER BY [CreateDate] desc;

4/Get skill by skill name

SELECT [SkillID]

,[SkillName]

,[CreateDate]

,[Description]

,[Status]

,[Img]

FROM [dbo].[Skill]

WHERE [SkillName] LIKE '%?%'

order by SkillID asc

offset ? row fetch next ? row only

#### i. Create CV of mentor

##### UI Design

#### 

| **Field Name** | **Field Type** | **Description** |
| --- | --- | --- |
| Avatar file | Button | Click to insert avatar picture |
| Full Name | Text Box | Click to enter full name of mentor |
| Account name | Text Box | Click to enter account name of mentor |
| Date of birth | Date Box | Click to enter date of birth of mentor |
| Email | Text Box | Click to enter email of mentor |
| Phone No. | Text Box | Click to enter phone number of mentor |
| Sex | Option Box | Click to enter sex of mentor |
| Address | Text Box | Click to enter address of mentor |
| Profession | Text Box | Click to enter profession of mentor |
| Framework | Text Box | Click to enter framework of mentor |
| Education | Text Box | Click to enter education of mentor |
| Price | Text Box | Click to enter mentor’s price base on their skill |
| Activity | Text Box | Click to enter activity of mentor |
| Profession Introduction | Text Box | Click to enter mentor’s information |
| Service description | Text Box | Click to enter information |
| Select skill | Check box | Click to add all skill mentor can teach |
| Achievement description | Text Box | Click to enter information |
| Save Change | Button | Click to create CV |
| Cancel | Button | Click to cancel to back to cv list |

##### Database access:

| **Table name** | **CRUD** | **Description** |
| --- | --- | --- |
| Mentor,CV | C,R | View current Mentor and CV of Mentor information |
| SkillList,Skill | C,R | View Mentor’s current skills |

##### SQL Commands:

1. Get Mentor info by MentorId

select\*from [dbo].[Mentor] where MentorID =?;

1. Get SkillList mà mentor không có

select s.SkillID,s.SkillName from Skill s left join SkillList sl on s.SkillID=sl.SkillID and sl.MentorID = ? where sl.SkillID is null;

1. Get email của mentor

select \* from [User] u join Mentor m on u.Username=m.Username where MentorID = ?

1. Insert CV

INSERT INTO [dbo].[CV]

([MentorID],

[Education],

[Experience],

[Activity],

[ProfessionIntroduction],

[Certificate],

[CreateDate],

[JobProfession],

[ServiceDescription],

[Status],

[Framework],

[Avatar],

[Price])

VALUES

(?, ?, ?, ?, ?, ?, GETDATE(), ?, ?, 'inactive', ?, ?, ?)

#### 

#### 

1. insertMentorSkills

INSERT INTO [dbo].[SkillList]

([SkillID],

[MentorID],

[Rating])

VALUES

(?, ?, null)

#### 

#### j. View profile-cv of mentor

##### UI Design

#### 

| **Field Name** | **Field Type** | **Description** |
| --- | --- | --- |
| Fullname | Text | Display full name of mentor |
| Account | Text | Display username of mentor |
| Job | Text | Display job of mentor |
| Introduction | Text | Display introduction of mentor |
| Service | Text | Display service of mentor |
| Achievement | Text | Display achievement of mentor |
| Statistic rating of each skill | Text,Slider | Display statistic rating of each skill by slider |
| Mentor rating star | Star | Display star base on average of rates |
| Rating & comment | Text,Star | Display rating star and comment of mentor |

##### Database access:

### 

| **Table name** | **CRUD** | **Description** |
| --- | --- | --- |
| Mentor,CV | Read | View current Mentor and CV of Mentor information |
| SkillList,Skill | Read | View Mentor’s current skills |

##### SQL Commands:

1. Get Mentor by Mentor ID

select\*from [dbo].[Mentor] where MentorID =?;

1. Get CV by Mentor ID if it is active

select\*from [dbo].[CV] where MentorID =? and Status='active';

1. Get Rate & Comment of Mentee by MentorID

select s.SkillID,s.SkillName,sl.Rating from Skill s join SkillList sl on s.SkillID=sl.SkillID where MentorID=?

1. Get statistic rating of each skill

select s.SkillID,s.SkillName,sl.Rating from Skill s join SkillList sl on s.SkillID=sl.SkillID

where MentorID=?

1. Get average rate of Mentor

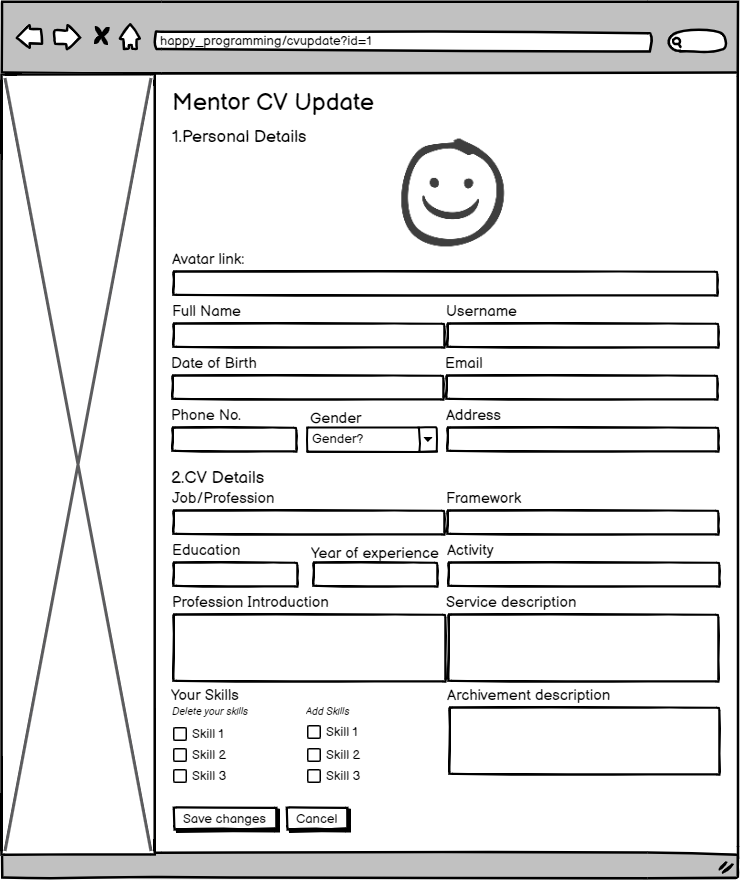
select Avg(Rate) from Rate where MentorID =?;

1. Get list of mentee

select\*from Mentee

#### k. Update CV

##### UI Design



| **Field Name** | **Field Type** | **Description** |
| --- | --- | --- |
| Avatar file | Button | Click to insert avatar picture |
| Full Name | Text Box | Click to enter full name of mentor |
| Account name | Text Box | Click to enter account name of mentor |
| Date of birth | Date Box | Click to enter date of birth of mentor |
| Email | Text Box | Click to enter email of mentor |
| Phone No. | Text Box | Click to enter phone number of mentor |
| Sex | Option Box | Click to enter sex of mentor |
| Address | Text Box | Click to enter address of mentor |
| Profession | Text Box | Click to enter profession of mentor |
| Framework | Text Box | Click to enter framework of mentor |
| Education | Text Box | Click to enter education of mentor |
| Price | Text Box | Click to enter mentor’s price base on their skill |
| Activity | Text Box | Click to enter activity of mentor |
| Profession Introduction | Text Box | Click to enter mentor’s information |
| Service description | Text Box | Click to enter information |
| Select skill | Check box | Click to add all skill mentor can teach |
| Achievement description | Text Box | Click to enter information |
| Save Change | Button | Click to create CV |
| Cancel | Button | Click to cancel action |

##### Database access:

### 

| **Table name** | **CRUD** | **Desription** |
| --- | --- | --- |
| Mentor,CV | Read,Update | View current Mentor’s information then update |
| SkillList,Skill | C,R,D | View Mentor’s current skill then add by create in skill list and remove by delete in skill list |

##### SQL Commands:

1. **Get mentor by ID: Display Mentor’s Personal details**

select\*from [dbo].[Mentor] where MentorID =?;

1. **Get CV by MentorId: Display Mentor’s CV detail**

select\*from [dbo].[CV] where MentorID =?;

1. **Update Mentor Personal details**

UPDATE [dbo].[Mentor]

SET [Username] =?, [Email] = ?, [Phone] = ?, [Address] = ?, [DateOfBirth] = ?, [FullName] = ?, [Gender] = ? WHERE MentorID=?

1. **Update Mentor’s CV Personal details**

UPDATE [dbo].[CV]

SET [Education] = ?, [Experience] = ?, [Activity] = ?, [ProfessionIntroduction] = ?, [JobProfession] = ?, [YearOfExperience] = ?, [ServiceDescription] = ?, [Framework] = ?,[Avatar] =? WHERE [MentorID] = ?

1. **Get list of Mentor’s skill**

select s.SkillID,s.SkillName from Skill s join SkillList sl on s.SkillID=sl.SkillID where MentorID=?

1. **Get list of skill Mentor don’t have**

select s.SkillID,s.SkillName from Skill s left join SkillList sl on s.SkillID=sl.SkillID and sl.MentorID = ?

where sl.SkillID is null;

1. **Delete Mentor Skills**

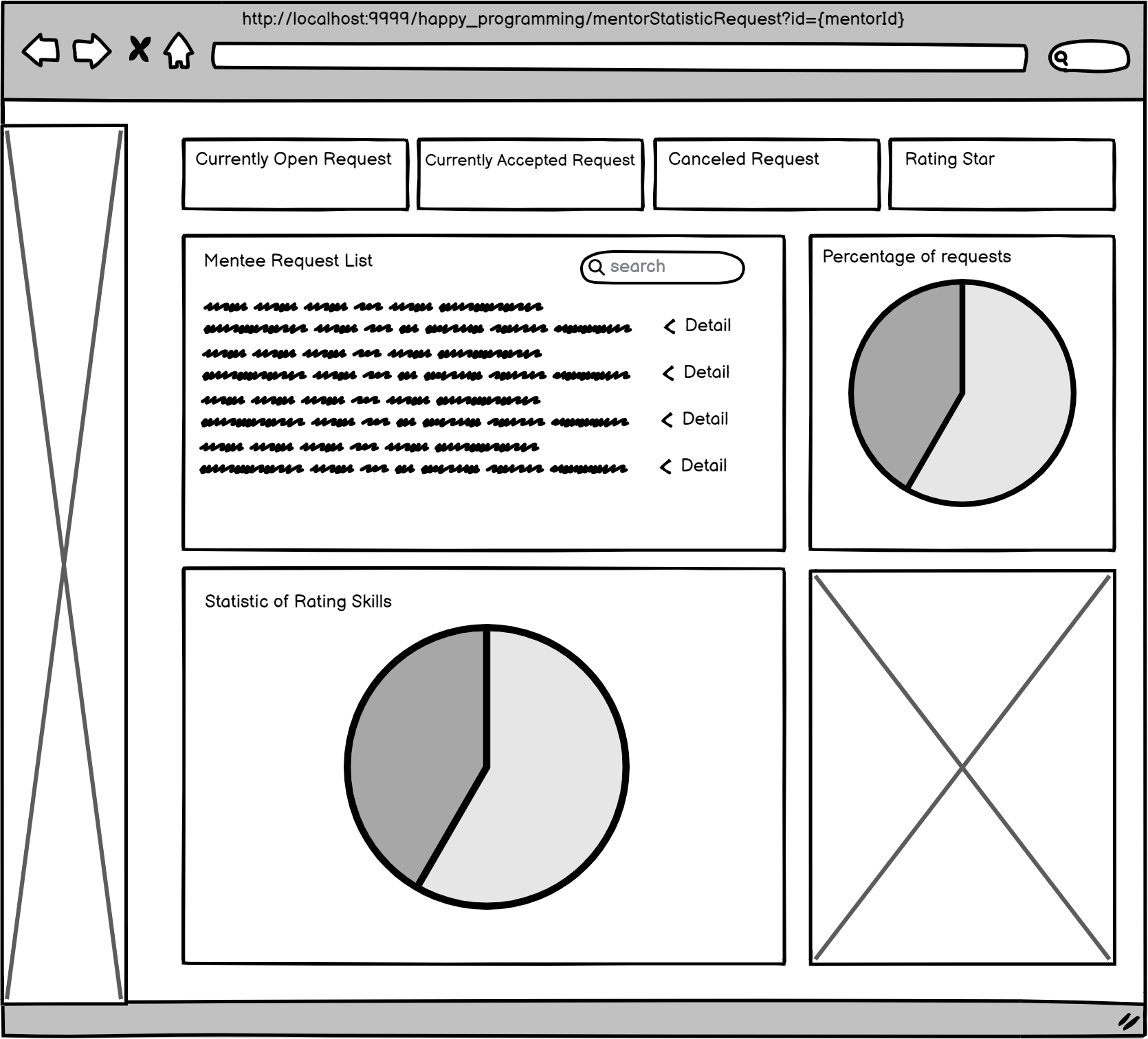
delete from SkillList where MentorID =? and SkillID =?

1. **Insert Mentor Skills**

INSERT INTO [dbo].[SkillList]([SkillID], [MentorID] ,[Rating]) VALUES (?,?,null);

#### l. View statistic request

##### UI Design

******

| **Field Name** | **Field Type** | **Description** |
| --- | --- | --- |
| Currently Open Request | Card | Display number of open request |
| Currently Accepted Request | Card | Display number of accepted request |
| Canceled Request | Card | Display number of canceled request |
| Rating Star | Card | Display number of average rating star |
| Search | Input box |  |
| STT | Table Column | Display number order of request |
| Title | Table Column | Display Title of request |
| Mentee | Table Column | Display Mentee name of request |
| Email | Table Column | Display Mentee Email of request |
| Start Time | Table Column | Display Start Time of request |
| Framework | Table Column | Display Framework of request |
| Price | Table Column | Display Price of request |
| Status | Table Column | Display Status of request |
| Action | Table Column | Display Action button of request |
| Detail | Button | Click to view detail of request |

##### *Database access:*

### 

| **Table name** | **CRUD** | **Desription** |
| --- | --- | --- |
| Request,Rate | Read | Count number of Open, Accepted, Canceled request and also request, rate average |
| Mentee,User | Read | Get list of Mentee, User to display mentee information |

##### SQL Commands:

1. **Count number of Open, Accepted, Canceled request and also request**

select count(MentorID) from Request where MentorID=? and Status ='Open';

select count(MentorID) from Request where MentorID=? and Status ='Pending';

select count(MentorID) from Request where MentorID=? and Status ='Canceled'

select count(MentorID) from Request where MentorID=?

1. **Get number of average of mentor**

select Avg(Rate) from Rate where MentorID =?;

1. **Get list of request by MentorId**

select \* from Request where MentorID=?

1. **Get list of Mentee**

select\*from Mentee;

1. **Get list of User**

select \* from [User]

#### 

#### m. Forget Password

##### UI Design

#### 

| **Field Name** | **Field Type** | **Description** |
| --- | --- | --- |
| Input required informations | Text box | Input registered email and account name to send reset password default |
| Submit | Button | Click to sent |

##### SQL Commands:

1/ Check if email exist

SELECT \* FROM [User] WHERE Email = ? AND Username = ?

2/Update new temporary password

BEGIN TRANSACTION;

UPDATE PasswordReset SET TemporaryPassword = ?

WHERE Username = ?;

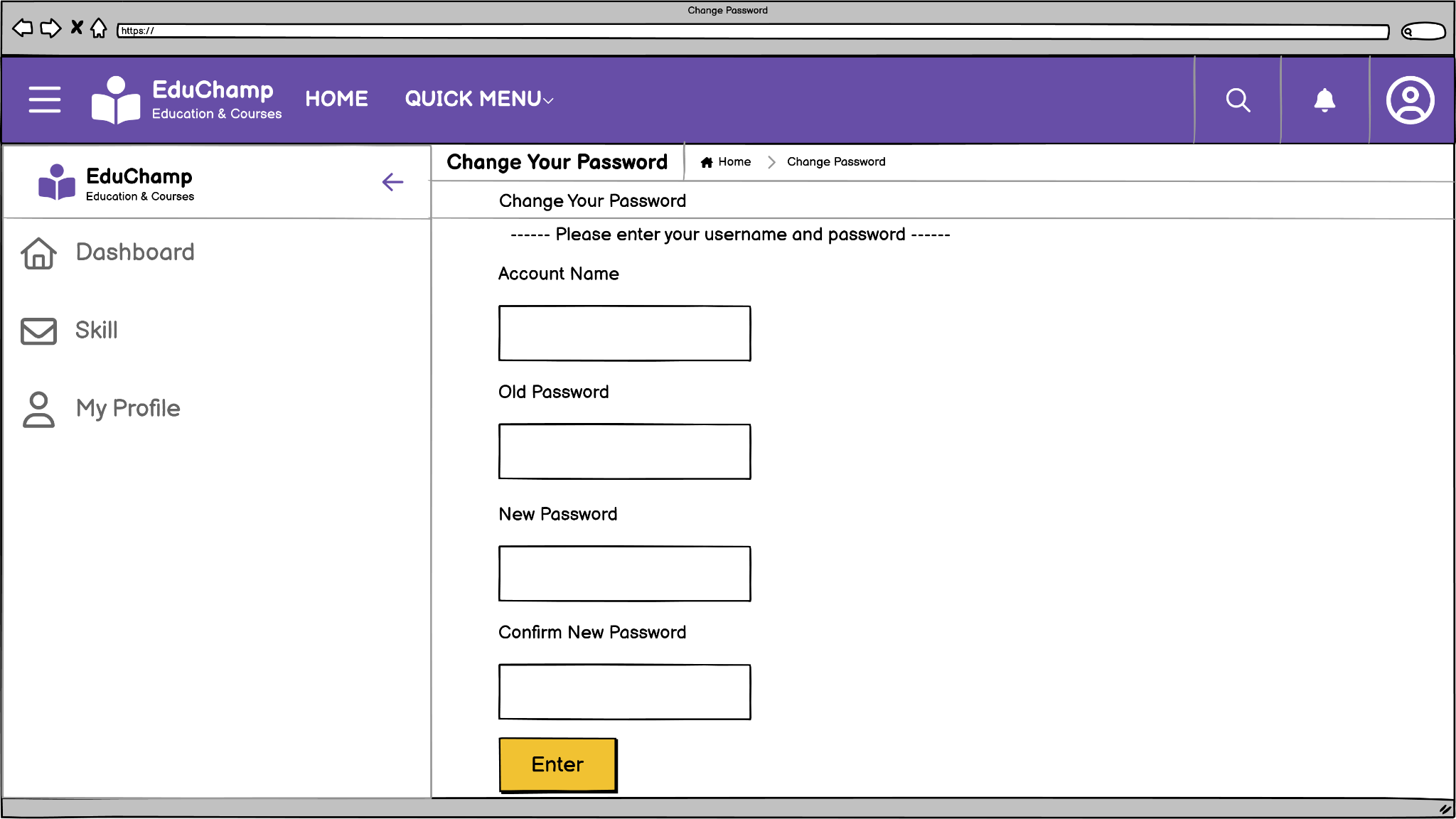
UPDATE [User] SET Password = ?

WHERE Username = ?;

COMMIT TRANSACTION;

#### n. Change Password

##### UI Design



| **Field Name** | **Field Type** | **Description** |
| --- | --- | --- |
| Input required informations | Text box | Input Account name, Old Password, New Password and Confirm New Password |
| Enter | Button | Click to change password |

##### SQL Commands:

1/ Check if username exist

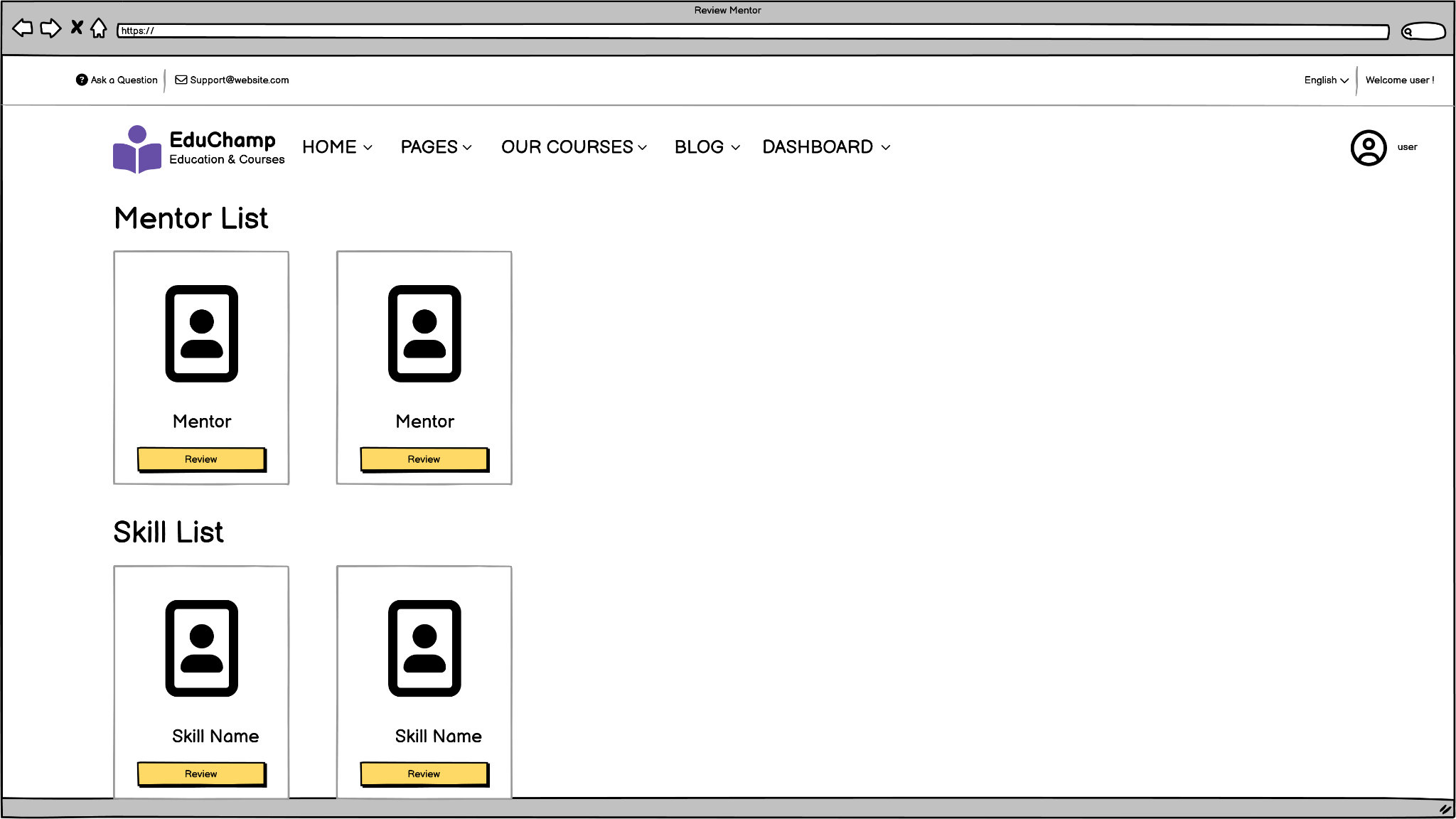
SELECT \* FROM [User] WHERE Username = ?

2/ Update password user

UPDATE [User] SET [Password] = ? WHERE Username = ?

#### o. Rate & Comment

##### UI Design



| **Field Name** | **Field Type** | **Description** |
| --- | --- | --- |
| Mentor list | Button | Description of mentor |
| Review | Button | Click to review mentor |

##### SQL Commands:

1/ List all mentor

SELECT [MentorID], [RoleID], [Username] , [CreateDate], [Phone], [Address], [DateOfBirth], [FullName], [Gender], [Status] FROM [Mentor]

2/ Get skill for completed or paid

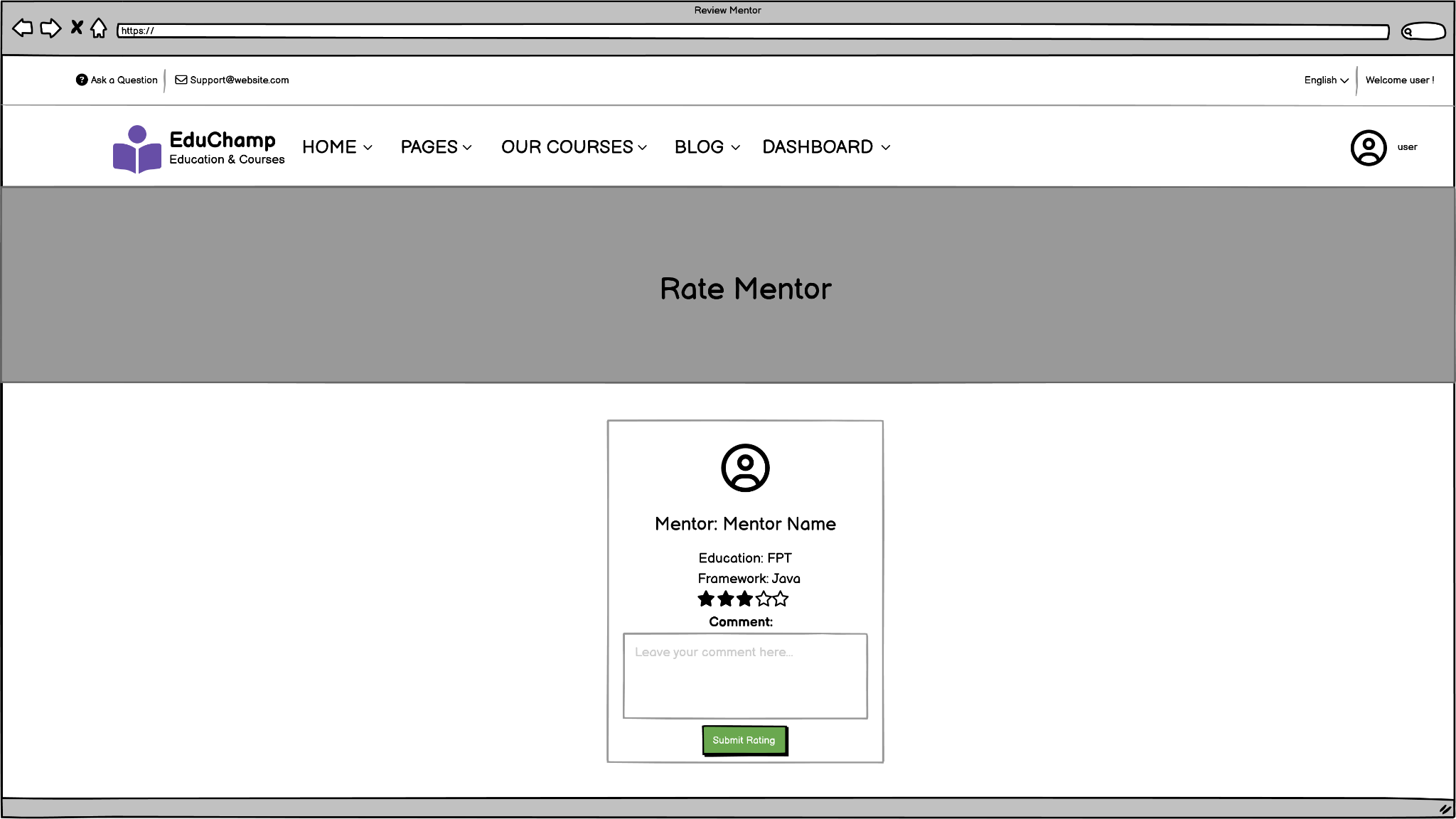
SELECT MIN(sl.SkillListID) AS SkillListID, sl.SkillID, sl.MentorID, MIN(sl.Rating) AS Rating, MIN(sl.CVID) AS CVID

FROM SkillList sl

JOIN Request r ON sl.SkillID = r.SkillID AND sl.MentorID = r.MentorID

WHERE r.MenteeID = ? AND (r.Status = 'Completed' OR r.Status = 'Paid')

GROUP BY sl.SkillID, sl.MentorID, r.Framework;



| **Field Name** | **Field Type** | **Description** |
| --- | --- | --- |
| Rate star | Radio | Rating the number of star for mentor |
| Comment | Text box | Leaving the review about that mentor |
| Submit Rating | Button | Saving mentee review |

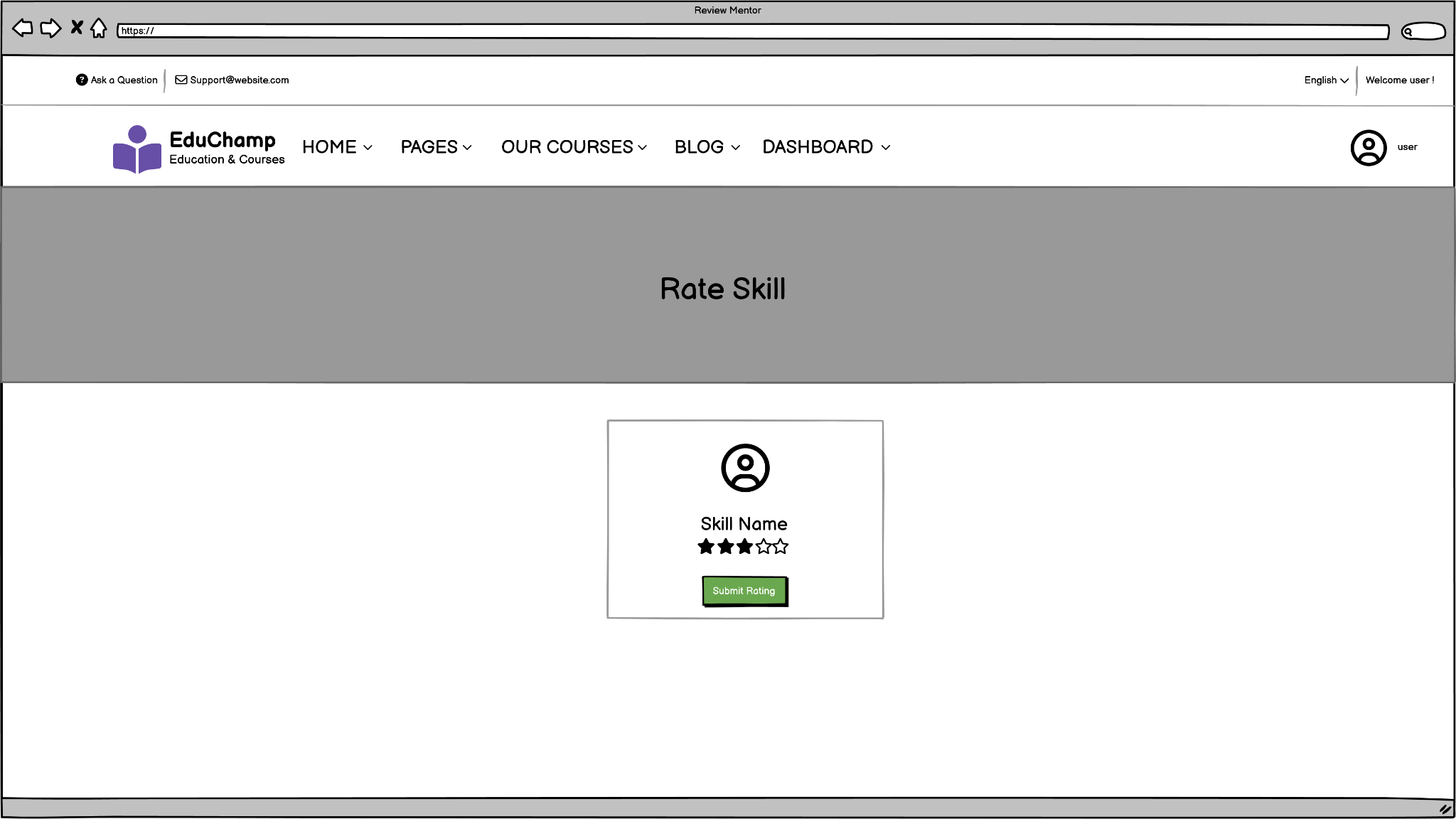
***SQL Commands:***

1/ Get mentee in session with username

SELECT [MenteeID], [RoleID], [Avatar], [Username], [CreateDate], [Phone], [Address], [DateOfBirth], [FullName], [Gender], [Status] FROM [Mentee] where Username = ?

2/ Get mentor in session with username

SELECT [MentorID], [RoleID], [Username], [CreateDate], [Phone], [Address], [DateOfBirth], [FullName], [Gender], [Status] FROM [Mentor] where Username = ?



| **Field Name** | **Field Type** | **Description** |
| --- | --- | --- |
| Rate star | Radio | Rating the number of star for skill |
| Submit Rating | Button | Saving mentee review |

***SQL Commands:***

1/ Rate skill

UPDATE SkillList SET Rating = ? WHERE SkillID = ?

#### p. Update Mentee Profile

##### UI Design



| **Field Name** | **Field Type** | **Description** |
| --- | --- | --- |
| Avatar file | Button | Upload image file from computer |
| Input information | Text box | Fill Mentee information to update |
| Save changes & Cancel | Button | Saving or cancel information |

##### SQL Commands:

1/ Get mentee in session with username

SELECT [MenteeID],[RoleID],[Avatar],[Username],[CreateDate],[Phone],[Address],[DateOfBirth],[FullName] ,[Gender] ,[Status]

FROM [dbo].[Mentee] where Username = ?

2/ Get mentor in session with username

SELECT [MentorID],[RoleID],[Username],[CreateDate],[Phone],[Address],[DateOfBirth],[FullName] ,[Gender] ,[Status]

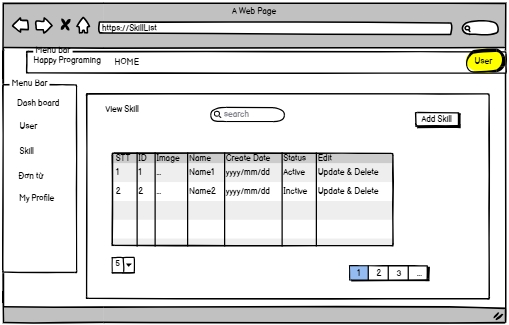
FROM [dbo].[Mentor] where Username = ?

3/ Update Mentee Profile

UPDATE Mentee SET RoleID = ?, Avatar = ?, Username = ?, CreateDate = ?, Phone = ?, Address = ?, DateOfBirth = ?, FullName = ?, Gender = ?, Status = ? WHERE username = ?

#### q. Admin view list of skill

**UI Design**



| **Field Name** | **Field Type** | **Description** |
| --- | --- | --- |
| ***Search Fields*** | | |
| Search | Text-box | This for admin to search skill by name |
| Add SKill | Link | This is for admin to go add new skill page |
| Skill Status | button | This is for the admin to switch Status which is Active or Inactive |
| 5 | Combo-box | This is for admin to chose number of skill display at one time |
| 1,2,3 pagination | link | This is for admin to go to next skill on next page |
| Delete | Link | Admin click to confirm delete Skill |
| Update | Link | Admin click to go to update skill chosen |

##### 

##### 

##### 

##### 

##### Database access:

### 

| **Table name** | **CRUD** | **Description** |
| --- | --- | --- |
| Skill | R,U,D | View all current skill on available on database Update status Skill Delete Skill by id |

##### SQL Command

1/ Select all skill

SELECT [SkillID]

,[SkillName]

,[CreateDate]

,[Description]

,[Status]

,[Img]

FROM [dbo].[Skill]  
**2/Delete skill by ID**DELETE FROM [dbo].[Skill]

WHERE SkillId = ?  
**3/ Update skill status**UPDATE [dbo].[Skill]

SET [Status] = ‘?’   
 WHERE SkillID = ?

#### r.Admin view list of Mentor

##### **UI Design**

| **Field Name** | **Field Type** | **Description** |
| --- | --- | --- |
| Search | Text-box | This is for admin search mentor by id,name, accountname, profession |
| Mentor Status | button | This is for the admin to switch Status which is Active or Inactive |
| 5 | Combo-box | This is for admin to chose number of mentor display at one time |
| 1,2,3 pagination | link | This is for admin to go to next skill on next page |

##### Database access:

| **Table name** | **CRUD** | **Description** |
| --- | --- | --- |
| Mentor | R,U | View all current mentor and update status |
| CV | R | View profession of Mentor |
| Request | R | Read all request |
| Rate | R | Read all rating for each mentor |

##### SQL Command

1/Select Mentor  
SELECT [MentorID]

,[RoleID]

,[Username]

,[CreateDate]

,[Email]

,[Phone]

,[Address]

,[DateOfBirth]

,[FullName]

,[Gender]

,[Status]

FROM [dbo].[Mentor]  
2/Select CV  
SELECT [CVID]

,[MentorID]

,[Education]

,[Experience]

,[Activity]

,[ProfessionIntroduction]

,[Certificate]

,[CreateDate]

,[JobProfession]

,[YearOfExperience]

,[ServiceDescription]

,[Status]

,[Framework]

,[Avatar]

FROM [dbo].[CV]

3/Select Rate  
SELECT [MentorID]

,[MenteeID]

,[CreateDate]

,[Status]

,[Comment]

,[Rate]

FROM [dbo].[Rate]

4/Update Mentor Status  
UPDATE [dbo].[Mentor]

SET [Status] = ‘?’

WHERE MentorID=’?’  
4/Get all request  
SELECT [RequestID]

,[MentorID]

,[MenteeID]

,[Price]

,[Note]

,[CreateDate]

,[Status]

,[Title]

,[DeadlineHour]

,[DeadlineDate]

,[Framework]

FROM [dbo].[Request]

5/Get all rate  
SELECT [MentorID]

,[MenteeID]

,[CreateDate]

,[Status]

,[Comment]

,[Rate]

FROM [dbo].[Rate]

6/ Get CV information   
WITH RankedCVs AS (

SELECT

CV.[CVID],

CV.[MentorID],

CV.[Education],

CV.[Experience],

CV.[Activity],

CV.[ProfessionIntroduction],

CV.[Certificate],

CV.[CreateDate],

CV.[JobProfession],

CV.[YearOfExperience],

CV.[ServiceDescription],

CV.[Status],

CV.[Framework],

CV.[Avatar],

ROW\_NUMBER() OVER (

PARTITION BY CV.[MentorID]

ORDER BY

CASE WHEN CV.[Status] = 'Active' THEN 1 ELSE 2 END, -- prioritize active CVs

CV.[CreateDate] DESC -- then by newest date

) AS rn

FROM [dbo].[CV]

)

SELECT

m.[MentorID],

r.[CVID],

r.[Education],

r.[Experience],

r.[Activity],

r.[ProfessionIntroduction],

r.[Certificate],

r.[CreateDate],

r.[JobProfession],

r.[YearOfExperience],

r.[ServiceDescription],

r.[Status],

r.[Framework],

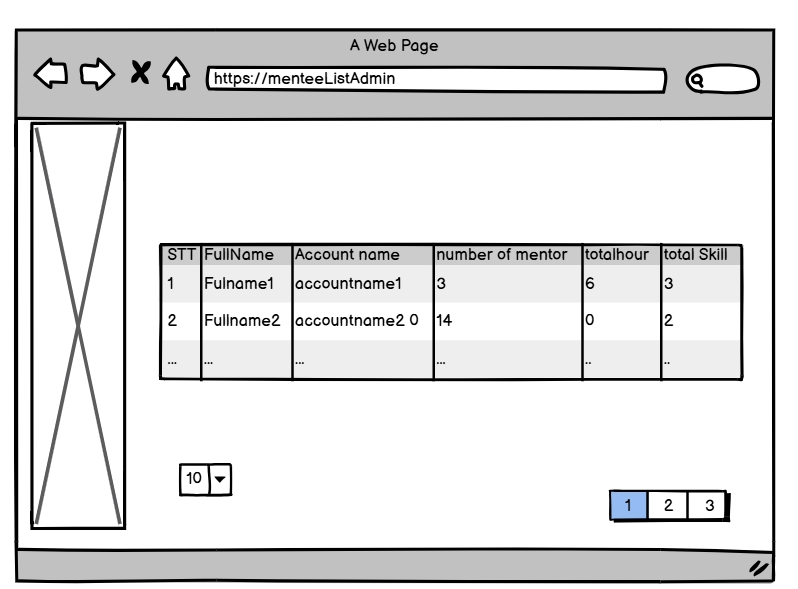
r.[Avatar]

FROM [dbo].[Mentor] m

LEFT JOIN RankedCVs r ON m.[MentorID] = r.[MentorID] AND r.rn = 1;

#### s.Admin view list of Mentee

##### UI Design



| **Field Name** | **Field Type** | **Description** |
| --- | --- | --- |
| 10 | Combo-box | This is for admin to chose number of skill display at one time |
| 1,2,3 pagination | link | This is for admin to go to next skill on next page |

##### Database access:

### 

| **Table name** | **CRUD** | **Description** |
| --- | --- | --- |
| Mentee | R | View all current mentee |
| Request | R | Count of mentor who mentee had and and studying Count of difference skill mentee have studied Get all valid request of mentee |
| Slot | R | Get all valid Slot |

##### SQL Command

1/Select Mentee pagination

SELECT [MenteeID]

,[RoleID]

,[Avatar]

,[Username]

,[CreateDate]

,[Email]

,[Phone]

,[Address]

,[DateOfBirth]

,[FullName]

,[Gender]

,[Status]

FROM [dbo].[Mentee]

order by FullName asc

OFFSET 0 ROWS

FETCH NEXT 10 ROW ONLY

2/Count mentor mentee have studied or studying  
select COUNT(DISTINCT MentorID) from Request

where [Status] != 'Cancel' and [Status] != 'Pending'

and MenteeID = ?

3/Count of difference skill mentee have studied  
select COUNT(DISTINCT SkillID) from Request

where [Status] != 'Cancel' and [Status] != 'Pending'

and MenteeID = ?

4/Get all valid request of mentee  
select COUNT(DISTINCT SkillID) from Request

where [Status] != 'Cancel' and [Status] != 'Pending'

and MenteeID = ?

5/Get all valid slot of each request   
SELECT s.[SlotID]

,[MentorID]

,[StartTime]

,[EndTime]

,[DayInWeek]

,[Status]

FROM [dbo].[Slot] s

right join RequestSlotItem rs

on s.SlotID = rs.SlotID

where rs.RequestID = ?

#### t.Admin view request

##### UI Design

#### 

#### 

| **Field Name** | **Field Type** | **Description** |
| --- | --- | --- |
| Search | text | Search request by id, username, title |
| Status | Combo-box | This for admin to find all status have specific status |
| Start | date | This for admin to choose start date want to filter |
| End | date | This for admin to choose end date want to filter |
| Submit | Submit | This is for admin to apply filter |
| ID | Link | This for admin to go to request detail page they click |
| 5 | Combo-box | This is for admin to chose number of skill display at one time |
| 1,2,3 pagination | link | This is for admin to go to next skill on next page |

##### Database access:

### 

| **Table name** | **CRUD** | **Description** |
| --- | --- | --- |
| Mentee | R | Get mentee who create the request |
| Request | R | Get all request ever created |

##### SQL Command

1/Select request pagination  
SELECT [RequestID]

,[MentorID]

,[MenteeID]

,[Price]

,[Note]

,[CreateDate]

,[Status]

,[Title]

,[Framework]

,[StartDate]

,[EndDate]

,[SkillID]

FROM [dbo].[Request]

order by RequestID

OFFSET ? ROWS

FETCH NEXT ? ROWS ONLY

2/Select mentee  
SELECT [MenteeID]

,[RoleID]

,[Avatar]

,[Username]

,[CreateDate]

,[Phone]

,[Address]

,[DateOfBirth]

,[FullName]

,[Gender]

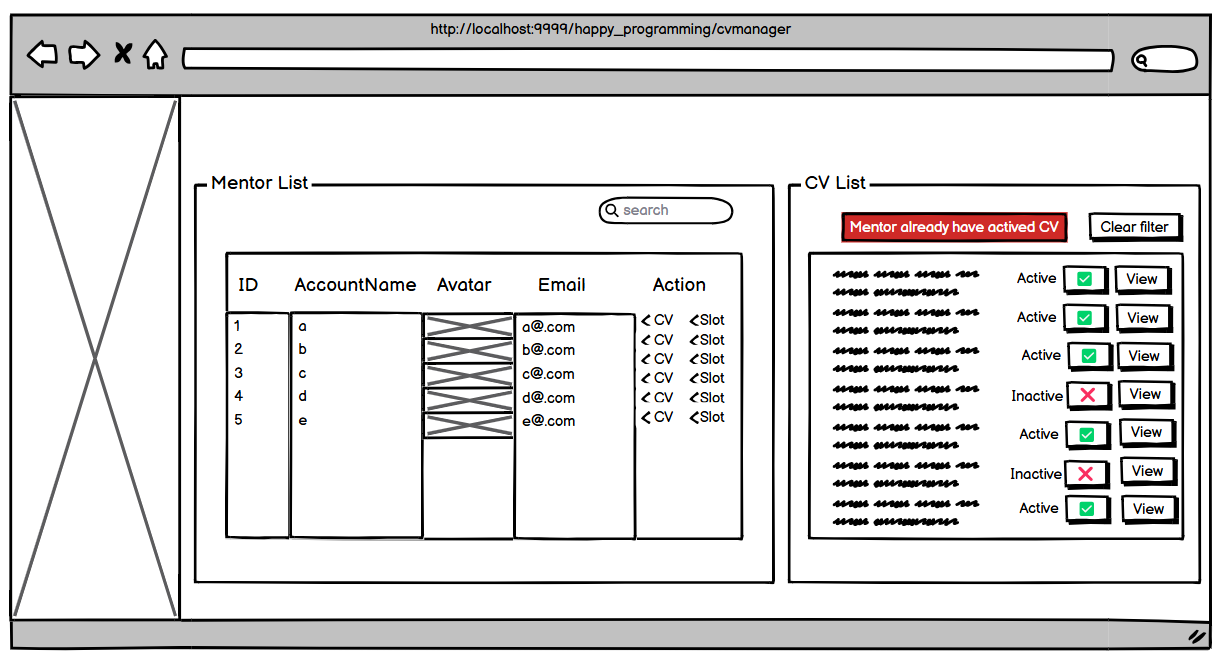
,[Status]

FROM [dbo].[Mentee]

where MenteeID = ?

#### u. Manager view CV & Slot list

##### UI Design

******

| **Field Name** | **Field Type** | **Description** |
| --- | --- | --- |
| Search | Text Box | Click to search mentor by name |
| id | Table Column | Display id of mentor |
| Account name | Table Column | Display Account name of mentor |
| Avatar | Table Column | Display avatar of mentor |
| Email | Table Column | Display email of mentor |
| Action | Table Column | Display action button of mentor |
| CV | Button | Click to category cv by mentor |
| Slot | Button | Click to category slot by mentor |
| Clear filter | Button | Click to reset search and category cv |
| (tick) icon | Status Button | Click to change status cv or slot of mentor to inactive |
| (x) icon | Status Button | Click to change status cv or slot of mentor to active |
| View | Button | Click to popup view detail cv or slot of mentor |

##### SQL Commands:

1. Get mentor and cv of mentor data

select \* from [Mentor]

select\*from CV

select \* from [Skill]

select \* from SkillList

select \* from [User]

1. **Get list active cv of mentor**

select\*from CV where MentorID=? order by CVID desc;

1. **Get list of Slot**

select \* from Slot where MentorID=? and Status = 'available'

select \* from Slot where MentorID=? and Status = 'inactive'

1. **Update status cv of mentor**

Update CV set Status ='active' where CVID =?

Update CV set Status ='inactive' where CVID =?

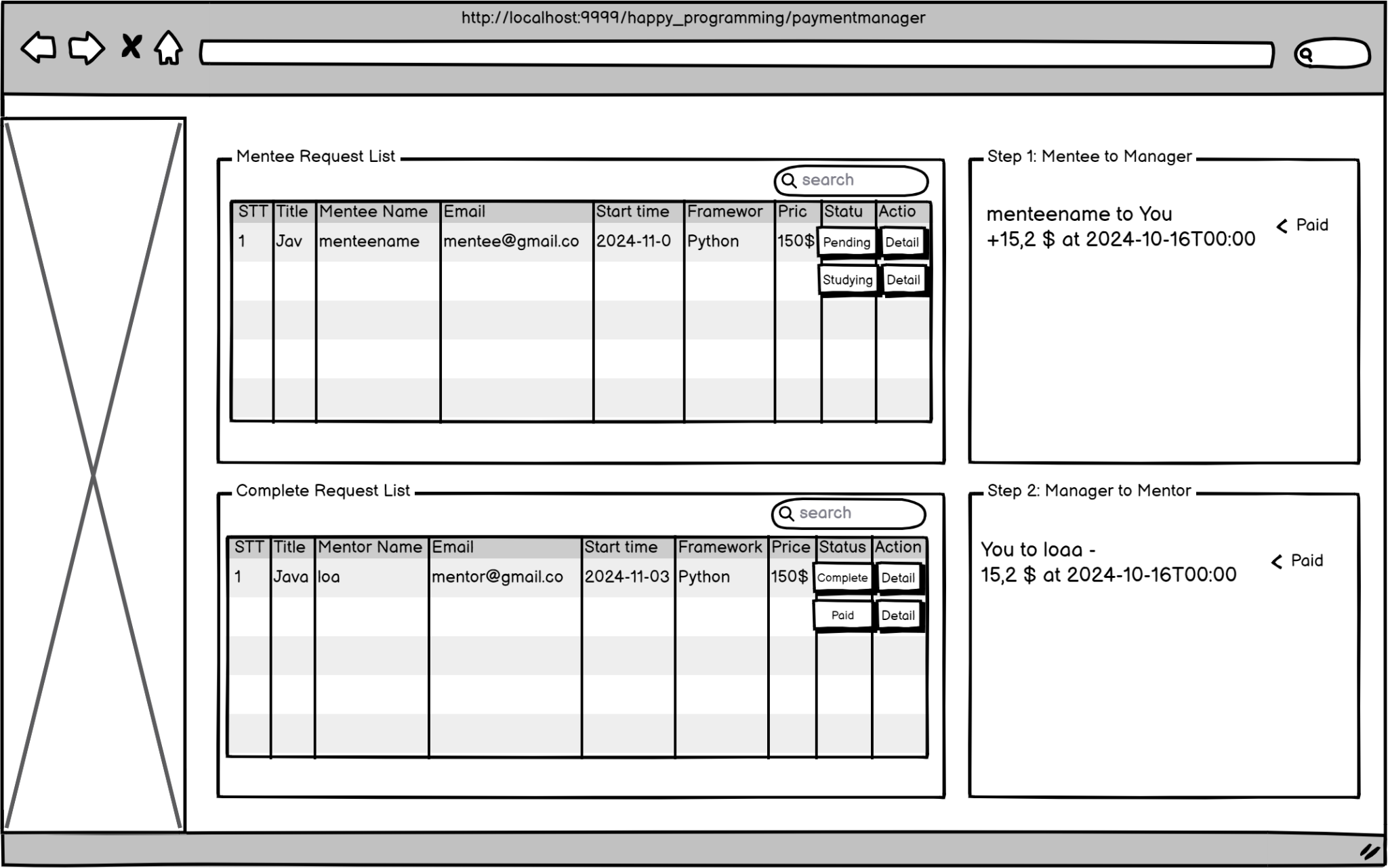
1. **Update status of Slot**

update Mentor set Status='inactive' where MentorID=?

update Mentor set Status='active' where MentorID=?

#### v. Manager payment request

##### UI Design

******

| **Field Name** | **Field Type** | **Description** |
| --- | --- | --- |
| Search | Text Box | Click to search mentor by name |
| STT | Table Column | Display STT of mentor |
| Mentee name | Table Column | Display Account name of mentee |
| Mentor name | Table Column | Display Account name of mentor |
| Email | Table Column | Display email of mentor/mentee |
| Start Time | Table Column | Display start time of request |
| Framework | Table Column | Display Framework of request |
| Price | Table Column | Display Price of request |
| Status | Table Column | Display Status of request |
| Action | Table Column | Display Action buttons of request |
| Pending | Button | Click to change status into Studying |
| Studying | Button | Click to change status into Pending |
| Complete | Button | Click to change status into Paid |
| Paid | Button | Click to change status into Complete |
| Detail | Button | Click to view payment detail |

##### SQL Commands:

1. **Get list of Request,Mentee,Mentor**

select \* from Request

select \* from Mentee

select \* from [Mentor]

select \* from [User]

1. **Get list of complete request**

select \* from Request where Status='Paid' or Status='Complete'

1. **Update status of Request from Pending to Studying and vice versa**

Update Request set Status ='Paid' where RequestID =?

Update Request set Status ='Complete' where RequestID =?

1. **Get request and payment list by request ID**

select \* from Request where RequestID =?;

select \* from Payment where RequestID =?;

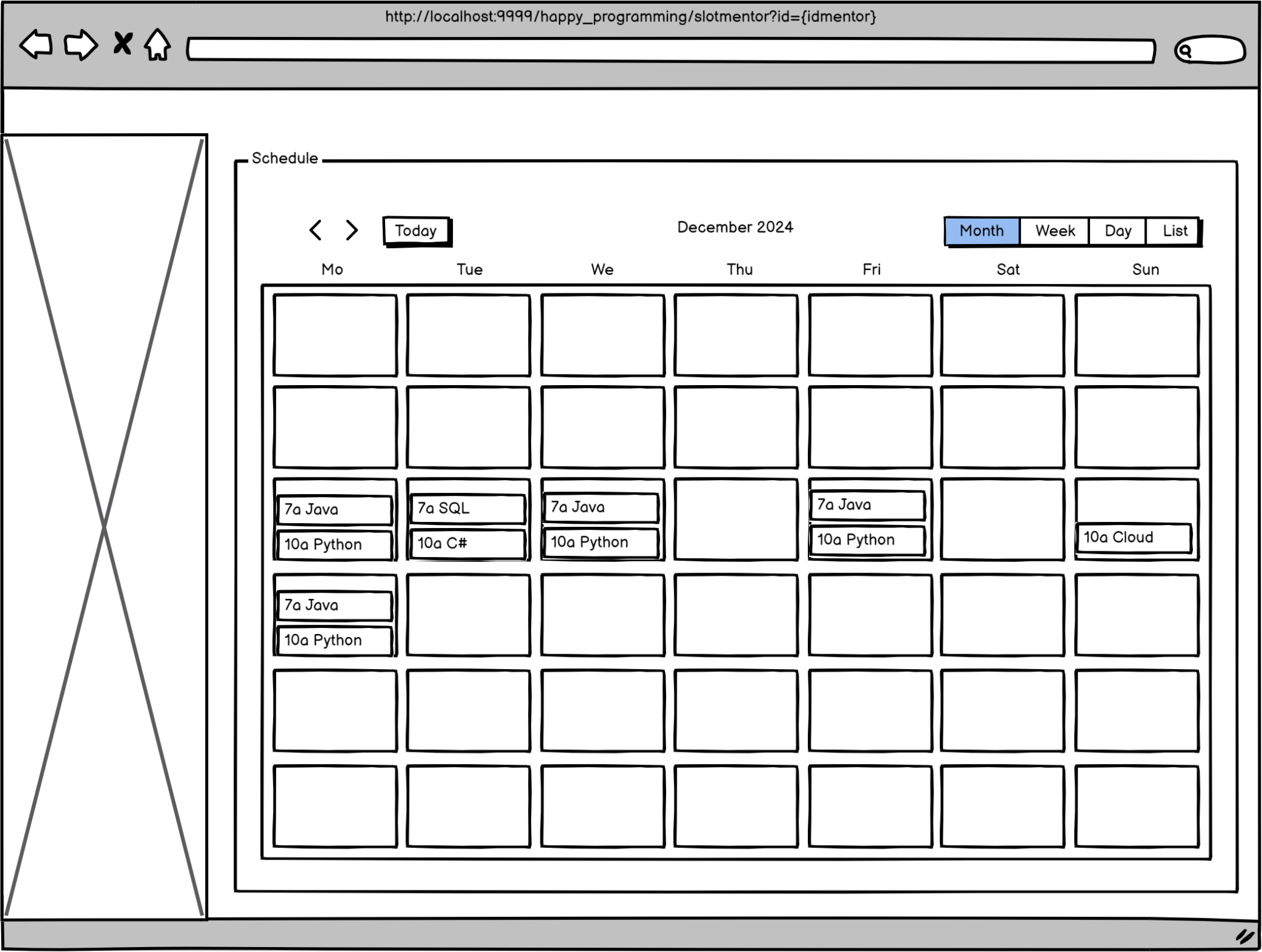
1. **Update status of Request from Complete to Paid and vice versa**

Update Request set Status ='Complete' where RequestID =?

Update Request set Status ='Paid' where RequestID =?

#### w. View slot

##### UI Design

******

| **Field Name** | **Field Type** | **Description** |
| --- | --- | --- |
| <>(nav button) | Nav Button | Click to view next date or pervious |
| Today | Button | Click to warp to today in schedule |
| Month | Button | Click to view month type schedule |
| Week | Button | Click to view week type schedule |
| Day | Button | Click to view day type schedule |
| List | Button | Click to view list type schedule |
| Schedule box | Button | Click to view that day type schedule |

##### SQL Commands:

1. **Get list of Slot by request of mentor**

select s.SlotID,s.StartTime,s.EndTime,s.DayInWeek,

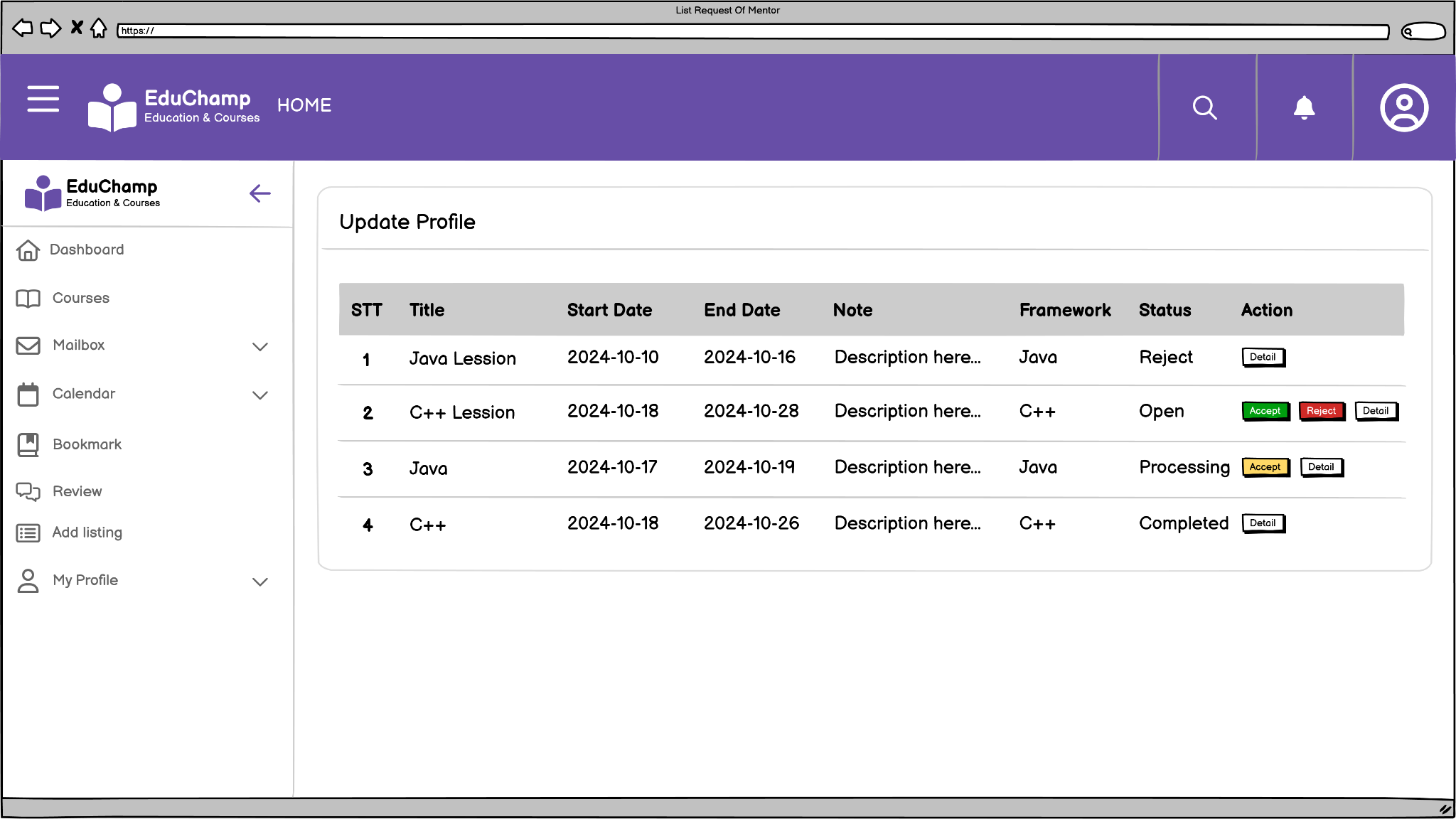
r.RequestID,r.MentorID,r.MenteeID,r.Price,r.Status,r.Title,r.Framework,r.StartDate,r.EndDate

from Slot s join RequestSlotItem rs on s.SlotID=rs.SlotID join Request r on rs.RequestID=r.RequestID

where r.MentorID=?

#### x. List request of mentor

##### UI Design

******

| **Field Name** | **Field Type** | **Description** |
| --- | --- | --- |
| Accept | Button | Click to accept request from Mentee |
| Reject | Button | Click to reject request from Mentee |
| Complete | Button | Click to finish the course |
| Detail | Button | Click to view detail of request |

##### SQL Commands:

1/ Get SlotId By RequestId

SELECT slotId FROM RequestSlotItem WHERE requestId = ?

2/Update Status By Mentor

UPDATE Request SET Status = ? WHERE RequestID = ?

3/ Reject Other Mentees For Same Slots

UPDATE Request SET Status = 'Reject'

WHERE RequestID IN (SELECT rsi.RequestID FROM RequestSlotItem rsi

JOIN Slot s ON rsi.SlotID = s.SlotID

WHERE rsi.SlotID IN (SELECT SlotID FROM RequestSlotItem WHERE RequestID = ?)

AND rsi.RequestID <> ? ) AND Status = 'Open'

4/ Update Slot Status To Unavailable

UPDATE Slot SET status = 'Unavailable' WHERE slotID = ?

5/ Update Slot Status To Available

UPDATE Slot SET status = 'Available' WHERE slotID = ?

#### y. List request by mentee

##### UI Design

******

| **Field Name** | **Field Type** | **Description** |
| --- | --- | --- |
| Update | Button | Click to accept request from Mentee |
| Cancel | Button | Click to reject request from Mentee |
| Detail | Button | Click to view detail of request |

##### SQL Commands:

1/ Get SlotId By RequestId

SELECT slotId FROM RequestSlotItem WHERE requestId = ?

2/Update Status By Mentor

UPDATE Request SET Status = ? WHERE RequestID = ?

3/ Reject Other Mentees For Same Slots

UPDATE Request SET Status = 'Reject'

WHERE RequestID IN (SELECT rsi.RequestID FROM RequestSlotItem rsi

JOIN Slot s ON rsi.SlotID = s.SlotID

WHERE rsi.SlotID IN (SELECT SlotID FROM RequestSlotItem WHERE RequestID = ?)

AND rsi.RequestID <> ? ) AND Status = 'Open'

4/ Update Slot Status To Unavailable

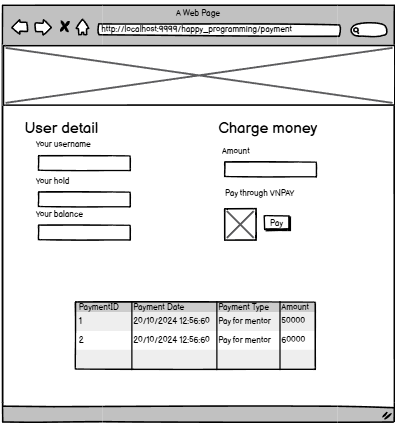
UPDATE Slot SET status = 'Unavailable' WHERE slotID = ?

5/ Update Slot Status To Available

UPDATE Slot SET status = 'Available' WHERE slotID = ?

#### z.Vnpay payment

##### UI Design



| **Field Name** | **Field Type** | **Description** |
| --- | --- | --- |
| Your username | Text Box | Show username |
| Your hold | Text box | Show user’s phone |
| Your balance | Text box | Show user’s wallet balance |
| Amount | Number box | Input money amount to charge |
| Pay | Button | Submit to create payment for recharging money |
| PaymentID | <p> | Show PaymentID |
| Payment Date | <p> | Show Payment Date |
| Payment Type | <p> | Show Payment Type |
| Amount | <p> | Show Payment Amount |

##### SQL command

**1/find mentee by username**

SELECT [MenteeID]

,[RoleID]

,[Avatar]

,[Username]

,[CreateDate]

,[Phone]

,[Address]

,[DateOfBirth]

,[FullName]

,[Gender]

,[Status]

FROM [dbo].[Mentee]

where Username = ?

2/Get wallet by user name

select \* from Wallet

where Wallet.Username = ?

3/Update wallet by username

UPDATE Wallet SET Balance = ? WHERE Username = ?

4/Create new payment

INSERT INTO payment (paymentDate, totalAmount, status, sender, receiver) VALUES (?, ?, ?, ?, ?, ?)

5/Get payment for mentee

SELECT\* FROM payment

WHERE sender = '?' AND status != 1;

6/Get payment for mentee pagination  
SELECT \* FROM payment WHERE sender = ? AND status != '1' ORDER BY paymentDate DESC OFFSET ? ROWS FETCH NEXT ? ROWS ONLY

#### 

#### 

#### 

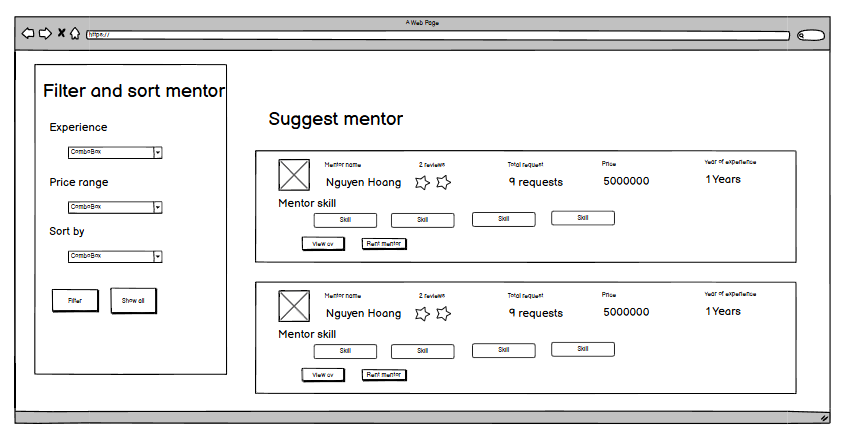
#### 

#### 

#### 

#### aa. List mentor suggest

##### UI Design



| **Field Name** | **Field Type** | **Description** |
| --- | --- | --- |
| Experience | Combo box | Select experience range of mentor |
| Price range | Combo box | Select price range of mentor |
| Sort by | Combo Box | Select sort by field |
| Filter | Button | Submit filter |
| Show all | Button | Show all mentor list by most request |
| Skill | <a> | Link to skill detail page of that skill |
| View cv | button | Link to page cv of mentor |
| Rent mentor | button | Redirect to mentor cv page |

##### Sql command:

1/

SELECT TOP 10 m.MentorID, COUNT(r.RequestID) AS RequestCount

FROM Mentor m

LEFT JOIN Request r ON m.MentorID = r.MentorID where m.Status='active'

GROUP BY m.MentorID

ORDER BY RequestCount DESC;

2/select Avg(Rate) from Rate where MentorID =?;

3/select \* from Rate where MentorID = ?

4/SELECT

RequestID, MentorID, MenteeID, Price, Note, CreateDate,Status,Title,Framework, StartDate, EndDate, SkillID

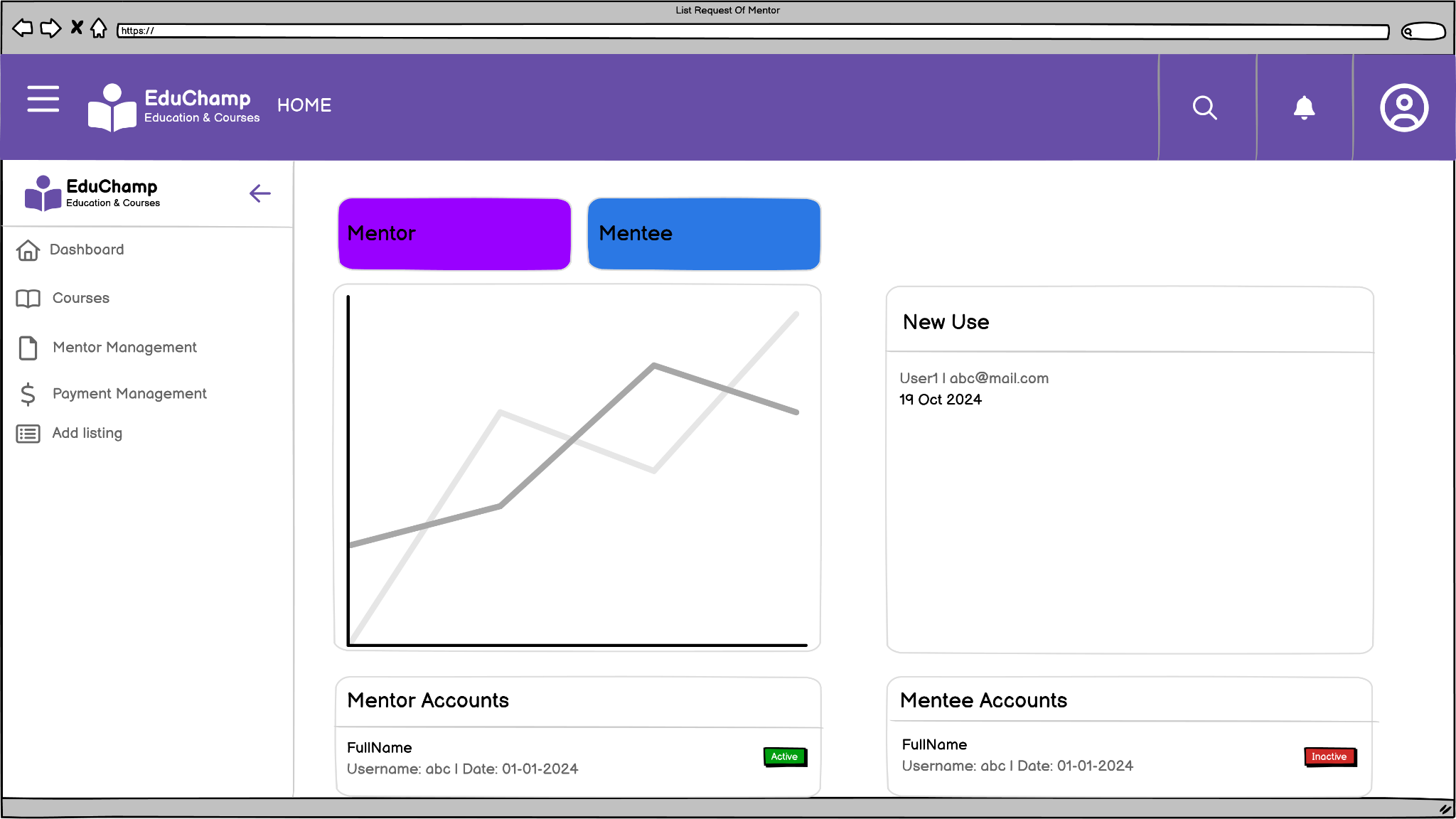
FROM dbo.Request

WHERE MentorID = ?;

5/select s.SkillID,s.SkillName from Skill s join SkillList sl on s.SkillID=sl.SkillID where MentorID=?

#### ab. Admin Dashboard

##### UI Design



| **Field Name** | **Field Type** | **Description** |
| --- | --- | --- |
| Active | Link | Admin click to active user |
| Inactive | Link | Admin click to inactive user |

##### 

##### 

##### 

##### 

##### Database access:

### 

| **Table name** | **CRUD** | **Description** |
| --- | --- | --- |
| User | R | View all current user on database |
| Mentor | R, U | View all current mentor on database Update status Mentor |
| Mentee | R, U | View all current mentee on database Update status Mentee |

SQL Command  
1/ Select all user  
SELECT \* FROM [User] WHERE RoleID NOT IN (3, 4) ORDER BY CreateDate DESC;

2/ Select all Mentor

SELECT [MentorID], [RoleID], [Username], [CreateDate], [Phone], [Address], [DateOfBirth], [FullName], [Gender], [Status]

FROM [dbo].[Mentor]

3/ Select all Mentee

SELECT [MenteeID], [RoleID], [Avatar], [Username], [CreateDate], [Email], [Phone], [Address], [DateOfBirth], [FullName], [Gender], [Status]

FROM [dbo].[Mentee]

4/ Get number of users through months

SELECT MONTH(CreateDate) AS month, YEAR(CreateDate) AS year, COUNT(\*) AS user\_count

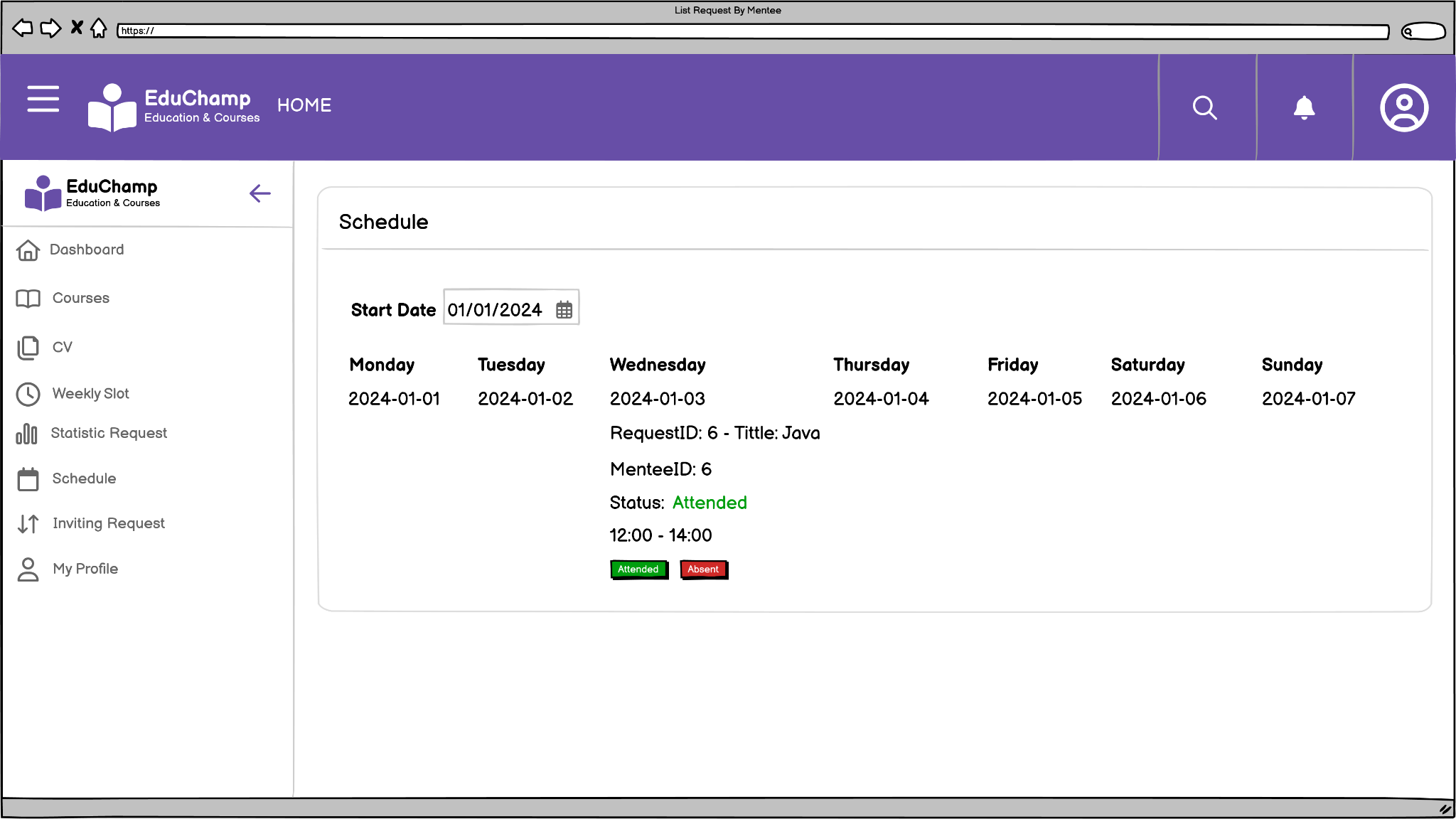
FROM [User]

GROUP BY YEAR(CreateDate), MONTH(CreateDate)

ORDER BY YEAR(CreateDate), MONTH(CreateDate);

#### ac. Mentor Schedule

##### UI Design



| **Field Name** | **Field Type** | **Description** |
| --- | --- | --- |
| Date | Button | Change date |
| Attended | Button | Mentor click to attended Mentee |
| Absent | Button | Mentor click to absent Mentee |

##### 

##### 

##### 

##### 

##### Database access:

### 

| **Table name** | **CRUD** | **Description** |
| --- | --- | --- |
| Attendance | R, U | View all attendance and update on database |

SQL Command  
1/ Get schedule by MentorID  
SELECT a.[AttendID], r.[RequestID], a.[slotDate], a.[Status], r.[MenteeID], s.[StartTime], s.[EndTime], r.[Title], s.[DayInWeek]

FROM [dbo].[Attendance] a

LEFT JOIN [RequestSlotItem] rs ON a.[RequestSlotItem] = rs.[RequestSlotItem]

LEFT JOIN [Slot] s ON s.[SlotID] = rs.[SlotID]

LEFT JOIN [Request] r ON r.[RequestID] = rs.[RequestID]

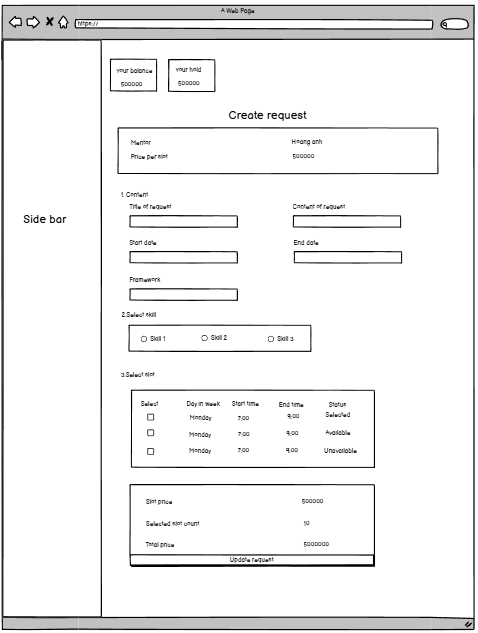
WHERE r.[MentorID] = ? AND a.[slotDate] BETWEEN ? AND ?

2/ Update status attendance

UPDATE [dbo].[Attendance] SET [Status] = ? WHERE [AttendID] = ?

#### ad. Update request

##### Ui design



| **Field Name** | **Field Type** | **Description** |
| --- | --- | --- |
| Tittle of request | Text Box | Input tittle of request from mentee to mentor |
| Content of request | Text box | Input content of request from mentee to mentor |
| Framework | Text Box | Input framework that mentee want to learn |
| Start Date | Timebox | Input start date that mentee want to support |
| End date | Date box | Input end date that mentee want to support |
| Skill | Checkbox | Allow user to select skill |
| Select | Checkbox | Allow user to select slots |
| Update request | Submit button | Submit all information that mentee input |

##### Sql command

1/Find mentor by id

SELECT [MentorID] ,[RoleID] ,[Username] ,[CreateDate] ,[Phone] ,[Address] ,[DateOfBirth],[FullName] ,[Gender] ,[Status]

FROM [dbo].[Mentor]

where MentorID = ?

2/Find cv by mentor id

select\*from [dbo].[CV] where MentorID =? and Status='active';

3/Get duplicate slot

SELECT \*

FROM RequestSlotItem r

JOIN Request rq ON r.RequestID = rq.RequestID

WHERE rq.Status = 'Open'

AND rq.MenteeID = ?

AND rq.MentorID = ?

AND rq.RequestID != ?;

4/Get list slot of mentor by mentor ID

SELECT SlotID, MentorID, StartTime, EndTime, DayInWeek, Status FROM Slot WHERE MentorID = ?

ORDER BY CASE DayInWeek

WHEN 'Monday' THEN 1

WHEN 'Tuesday' THEN 2

WHEN 'Wednesday' THEN 3

WHEN 'Thursday' THEN 4

WHEN 'Friday' THEN 5

WHEN 'Saturday' THEN 6

WHEN 'Sunday' THEN 7

ELSE 8 END";

5/Get list skill of mentor

select s.SkillID,s.SkillName from Skill s left join SkillList sl on s.SkillID=sl.SkillID and sl.CVID = ?

6/Get request by requestID:

SELECT \* FROM [dbo].[Request] WHERE RequestID = ?

7/Get user wallet

SELECT \* FROM Wallet WHERE Username = ?

8/Update payment amount

UPDATE payment SET TotalAmount = ? WHERE requestId = ?

9/ Delete skill selected

DELETE FROM [dbo].[RequestSlotItem]

WHERE RequestID = ?;

10/Add new selected skill

INSERT INTO [dbo].[RequestSlotItem] (RequestID, SlotID)

VALUES (?, ?);

11/Update request

UPDATE [dbo].[Request]

SET [MentorID] = ?,

[MenteeID] = ?,

[Price] = ?,

[Note] = ?,

[CreateDate] = ?,

[Status] = ?,

[Title] = ?,

[Framework] = ?,

[StartDate] = ?,

[EndDate] = ?,

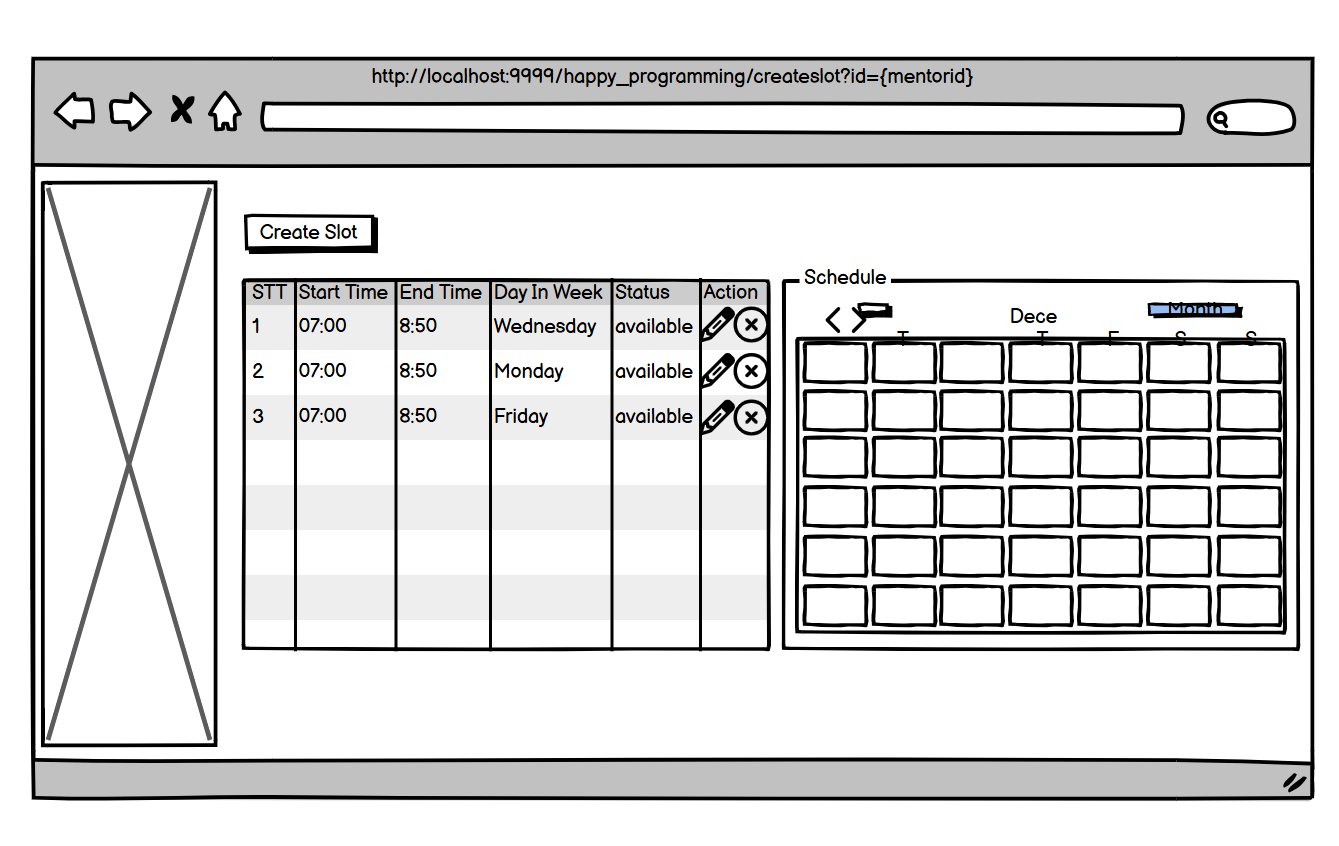
[SkillID] = ?

WHERE [RequestID] = ?;

##### 

#### ae.**CRUD Slot of Mentor**

##### UI Design

******

| **Field Name** | **Field Type** | **Description** |
| --- | --- | --- |
| Create Slot | Button | Click to create request |
| STT | Table Column | Display numerical order of slot |
| Start Time | Table Column | Display start time of slot |
| End Time | Table Column | Display end time of slot |
| Day In Week | Table Column | Display day in week of slot |
| Status | Table Column | Display status of slot |
| Action | Table Column | Display action of slot |
| (Pencil icon) | Button | Click to update request |
| (X icon) | Button | Click to delete request |
| Schedule | Calendar | Display slot of mentor |

##### *Database access:*

### 

| **Table name** | **CRUD** | **Description** |
| --- | --- | --- |
| Slot | C,R,U,D | Create slot, Readd slot, Update slot, Delete slot |
| Mentor | Read | Get mentor information |

##### SQL Commands:

1. **Add slot**

select\*from [dbo].[Mentor] where MentorID =?;

select \* from Slot where MentorID=? and Status = 'available'

select \* from Slot where MentorID=? and Status = 'inactive'

1. **Delete slot**

SELECT \* FROM Slot WHERE SlotID = ?

select \* from Slot where MentorID=? and Status = 'available'

INSERT INTO [dbo].[Slot] ([MentorID],[StartTime],[EndTime],[DayInWeek],[Status],[CVID])

VALUES (?,?,?,?,?,?)

DELETE FROM [dbo].[Slot] WHERE SlotID=?

1. **Update slote**

select top (1) \* from [dbo].[CV] where MentorID =? order by CreateDate desc;

# IV. Non-Functional Requirement

# 

| **Category** | **Description** |
| --- | --- |
| Performance Efficiency | **Transaction Per Second (TPS) / Concurrent Users (CCU)**  The system must be able to handle 100 TPS/100 CCUs for all features |
| Performance Efficiency | **Processing Time**  When moving from one screen to another, the system must respond to an outcome within 5 seconds on the 1st load. 2nd load onwards this should run faster due to caching mechanisms (3 Seconds) |
| Scalability & Data Volume | The system must be able to handle 10% increase in volume per year, following data forecast. |
| Audit trail | Ensure using technical fields to trail user actions: Created on, Created by, Updated on, Updated by. |
| Localization | The system must support English (British) and Vietnamese |
| Usability | All deletions to have a confirm (Y/N) for the delete. For web application, confirmation dialog shall be the browser default one |
| Usability | When navigating away from added/updated data, the system will prompt a confirm (Y/N) to save the data. For web application, confirmation dialog shall be the browser default one |
| Usability | All error messages should try to give suggestions of what the error is and if possible, solution(s) to rectify the problem |
| Security | Before accessing the system, users will authenticate themselves with a username and password |
| Security | All sensitive data stored will be encrypted (Password |
| Online Help | No on-line User Documentation or contextual Help will be provided. |
| Maintainability | All documentation and coding must follow naming conventions defined in our coding standards |
| Maintainability | The system should implement monitoring and alerting to support issue triaging and troubleshooting |

# V. Appendix

## 1. Common Error Definition

| **ID** | **Common Error Definition** |
| --- | --- |
| ER500 | Cannot connect to database |
| ER404 | Page not found |