



دانشگاه اصفهان

دانشکده مهندسي کامپیوتر

گروه مهندسی معماری سيستم هاي کامپيوتر

گزارش پروژه کارشناسی

رشته‌ مهندسی معماری سيستم هاي کامپيوتر

عنوان پروژه:

پیاده‌سازی یک سامانه‌ی مدیریت آموزش

استاد راهنما:

دکتر کمال جمشیدی

پژوهشگر:

مهدی جعفری

دی 1402

دانشگاه اصفهان

دانشکده مهندسي کامپیوتر

گروه مهندسی معماری سيستم هاي کامپيوتر

پروژه کارشناسی رشته‌ي مهندسی کامپيوتر گرايش سخت‌افزار

آقای مهدی جعفری

تحت عنوان

پیاده‌سازی یک سامانه‌ی مدیریت آموزش

در تاريخ / / 13 توسط هيأت داوران زير بررسي و با نمره به تصويب نهايي رسيد.

1- استاد راهنماي پروژه:

دکتر کمال جمشیدی امضا

2- استاد داور :

دکتر امضا

امضاي مدير گروه

تشکر و قدرداني

بدین‌وسیله تشکر قلبی خود را از استاد عالی‌قدر جناب آقای دکتر کمال جمشیدی که در تمام مسیر، دلسوزانه راهنمایی این پایان‌نامه را عهده‌دار بودند و از هیچ کمکی دریغ ننمودند، ابراز می‌کنم و توفیقات روزافزون برای ایشان آرزو دارم.

همچنین از داوران گرامی که زحمت داوری این پایان‌نامه را عهده‌دار هستند کمال تشکر و قدردانی را دارم.

تقديم به

محضر گرامی خانواده‌ی مهربانم که همواره بزرگ‌ترین سرمایه‌ی من بوده و هستند و در تمام مسیر زندگی با فداکاری خود، خوب زیستن را به من آموخته‌اند.

چکيده:

این پروژه تحت عنوان پیاده‌سازی یک سامانه‌ی مدیریت آموزشی انجام شده است. اگرچه چنین‌ سامانه‌ای به صورت مشابه موجود است، اما سامانه‌ی جدید امکانی برای پردازش تصویر و نظارت بر فعالیت کلاس‌ها ایجاد کرده است که مزیت اصلی آن محسوب می‌شود. لازم به ذکر است که در ابتدا، اتصال این سامانه به سایت گلستان نیز یکی اهداف پروژه بود که به دلیل ضیق وقت و نیاز به هماهنگی با تیم توسعه‌ی گلستان، محقق نشد. البته باتوجه به نیاز حال حاضر، توسعه‌ی این پروژه و ادامه‌دادن این مسیر بسیار توصیه می‌شود.

این سامانه با استفاده از زبان جاوا و فریم‌ورک Spring Boot، و در قالب دو برنامه‌ی متصل نوشته شده است. در این برنامه‌ها از ابزار Thymleaf برای ارتباط با بخش ظاهری (front-end) بهره گرفته شده است. همچنین بخش ظاهری آن با استفاده از زبان‌های HTML[[1]](#footnote-2)، CSS[[2]](#footnote-3)، JavaScript و فریم‌ورک Bootstrap و کتابخانه‌ی jQuery نوشته شده است. یکی از این دو برنامه بر روی سرور و برنامه دیگر بر روی سیستم‌های هر کلاس و یا آزمایشگاه اجرا می‌شود و برای مشاهده تمام عملکرد سامانه، اجرای هم‌زمان این دو الزامی است.

این سامانه به صورت کاملاً متن‌باز بر روی Github قرار داده شده است و امکان مشاهده تمام فرایند پیشرفت آن فراهم است. در این سامانه قابلیت ثبت‌نام و ورود دانشجویان، تعریف اساتید و کلاس‌های مختلف و همچنین قابلیت قراردادن اطلاعیه، امتحان و تکلیف در هر کلاس وجود دارد. علاوه بر ویژگی‌های مذکور، با استفاده از پردازش تصاویر دریافتی از هر کلاس به صورت بلادرنگ، امکان تشخیص حضور افراد در کلاس میسر است. این پردازش تصویر به کمک کتابخانه‌ی قدرتمند OpenCV[[3]](#footnote-4) که عموماً در سیستم‌های بینایی ماشین‌های بلادرنگ (real-time computer vision) کاربرد دارد، اجرا شده است.

واژگان کليدي: سامانه‌ی مدیریت آموزش، استفاده از پردازش تصویر در زبان جاوا، بینایی ماشین بلادرنگ، کتابخانه‌ی OpenCV در جاوا

[فصل اول مقدمه 6](#_Toc156845545)

[1-1- هدف پروژه 6](#_Toc156845546)

[1-2- کاربردهای پروژه 6](#_Toc156845547)

[1-3- ساختار پایان نامه 7](#_Toc156845548)

[فصل دوم مفاهیم اولیه 8](#_Toc156845549)

[2-1- مقدمه 8](#_Toc156845550)

[2-2- ایجاد پروژه 8](#_Toc156845551)

[2-3- ایجاد وابستگی‌های برنامه 8](#_Toc156845552)

[2-3- بهره‌گیری از Thymleaf 9](#_Toc156845553)

[2-4- دیتابیس 10](#_Toc156845554)

[2-4-1- MySQL 10](#_Toc156845555)

[2-4-2- XAMPP 10](#_Toc156845556)

[2-5- کنترل ورژن Git 11](#_Toc156845557)

[2-6- جمع‌بندي 11](#_Toc156845558)

[فصل سوم شرح کلیات پروژه 12](#_Toc156845559)

[3-1- مقدمه 12](#_Toc156845560)

[3-2- کلاس‌ها 12](#_Toc156845561)

[3-2-1- Entity 12](#_Toc156845562)

[3-2-2- Jpa Repositories 13](#_Toc156845563)

[3-2-3- Controllers 14](#_Toc156845564)

[3-2-4- Carriers 16](#_Toc156845565)

[3-3- طراحی 16](#_Toc156845566)

[3-3-1- ساختار 18](#_Toc156845567)

[3-4- اعتبارسنجی 19](#_Toc156845568)

[3-5- جمع‌بندي 20](#_Toc156845569)

[فصل چهارم پردازش تصویر 21](#_Toc156845570)

[4-1- مقدمه 21](#_Toc156845571)

[4-2- معرفی کاربری 21](#_Toc156845572)

[4-3- معرفی عملکرد برنامه 22](#_Toc156845573)

[4-3-1- شروع و توقف نظارت 22](#_Toc156845574)

[4-3-2- تابع سازنده‌ی کلاس MotionDetector 23](#_Toc156845575)

[4-3-3- تابع startDetecting 23](#_Toc156845576)

[4-3-4- تابع sendStatus 23](#_Toc156845577)

[4-3-5- تابع setStatus 24](#_Toc156845578)

[4-4- جمع‌بندي 25](#_Toc156845579)

[فصل پنجم چالش‌ها و نتیجه‌گیری 26](#_Toc156845580)

[5-1- چالش‌ها و راه‌حل‌های آن‌ها 26](#_Toc156845581)

[5-1-1- Id generation 26](#_Toc156845582)

[5-1-2- جابجایی میان توابع Controller به همراه مقادیر 27](#_Toc156845583)

[5-1-3- OpenCV 28](#_Toc156845584)

[5-2- نتیجه‎گیری 28](#_Toc156845585)

[پيوست 1: لینک گیت‌هاب پروژه‌ها 29](#_Toc156845586)

[منابع 30](#_Toc156845587)

[شکل 2-1: نحوه‌ی کانفیگ دیتابیس در APPLICATION.PROPERTIES 10](#_Toc156845495)

[شکل 2-2: نمای کلی دیتابیس برنامه با استفاده از برنامه XAMPP 11](#_Toc156845496)

[شکل 3-1: Controller مربوط به داشبورد برنامه 16](#_Toc156845497)

[شکل 3-2: یک نمونه پیغام Toast نمایش داده شده در سامانه 16](#_Toc156845498)

[شکل 3-3: صفحه‌ی ثبت‌نام برنامه، طراحی شده با Thymeleaf 17](#_Toc156845499)

[شکل 3-4: پوشه‌بندی بخش ظاهری برنامه 18](#_Toc156845500)

[شکل 3-5: ساختار تابع اعتبار سنجی 19](#_Toc156845501)

[شکل 4-1: صفحه مخصوص کلاس، قبل از شروع نظارت 22](#_Toc156845502)

[شکل 4-2: فرایند شروع و پایان نظارت 23](#_Toc156845503)

[شکل 4-3: محتویات تابع ارسال کننده RestTemplate 24](#_Toc156845504)

[شکل 4-4: تابع تمیز دهنده کلاس‌های فعال و غیرفعال 24](#_Toc156845505)

[شکل 4-5: شیوه‌ی نمایش کلاس‌های فعال و غیر‌فعال در سامانه 25](#_Toc156845506)

[شکل 5-1: نمونه استفاده‌ی @GeneratedValue در پروژه 27](#_Toc156845507)

[شکل 5-2: یک نمونه استفاده‌ی RedirectView در پروژه 27](#_Toc156845508)

[جدول 2-1: وابستگی‌های خارجی موردنیاز برای پروژه 9](#_Toc156845244)

[جدول 3-1: هر Entity و کاربرد آن در پروژه 13](#_Toc156845245)

[جدول 3-2: هر Controller و کاربرد آن در پروژه 15](#_Toc156845246)

مخفف‌ها:

|  |  |
| --- | --- |
| HTML  CSS  OpenCV  IDE  DOM  AJAX | HyperText Markup Language  Cascading Style Sheets  Open Computer Vision  Integrated development environment  Document Object Model  Asynchronous JavaScript and XML |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

# فصل اول مقدمه

## 1-1- هدف پروژه

هدف این پروژه، ارائه‌ی یک سامانه‌ی به‌روز برای مدیریت کلاس‌ها و آزمایشگاه‌ها است تا کاربران بتوانند در کمترین زمان، از امکاناتی مانند ثبت‌نام و ورود دانشجویان، تعریف اساتید و کلاس‌های مختلف و همچنین قابلیت قراردادن اطلاعیه، امتحان و تکلیف در هر کلاس، بهره‌مند گردند. علاوه بر اهداف مذکور، هدف اصلی و نوین این پروژه، تشخیص حضور افراد در هر کلاس و یا آزمایشگاه است که به صورت یک سیستم بینایی ماشین بلادرنگ پیاده‌سازی شده است. این سامانه همچنین امکان ذخیره‌ی تصویر افراد حاضر در کلاس را نیز میسر می‌کند. خروجی این سامانه بر روی مرورگر کاربران قابل‌استفاده است.

## 1-2- کاربردهای پروژه

در عصری زندگی می‌کنیم که بسیاری از رفتارهای تکراری و خسته‌کننده، توسط هوش مصنوعی و ابزارهای جدید جایگزین می‌شوند و باارزش نهادن به زمان افراد، موج عظیمی از فعالیت‌ها به کامپیوترها سپرده خواهند شد. یکی از اصلی‌ترین نیازهای موجود در مجموعه کلاس‌های هر دانشگاه، فقدان اطلاعات درباره پر یا خالی بودن کلاس‌های آن مجموعه است. این اطلاعات معمولاً با پرسش‌وپاسخ از مسئولین مرتبط و یا سرکشی حضوری به محل موردنظر حاصل می‌شوند. درحالی‌که چنین رفتاری در عصر اطلاعات پذیرفتنی نیست. این سامانه علاوه بر ارائه خدمات سامانه‌های سابق، این تقاضا را نیز به‌راحتی برآورده می‌کند.

باتوجه‌به موارد گفته شده، می‌توان کاربرد بسیاری برای بخش پردازش تصویر این سامانه در مجموعه‌های آموزشی، سالن‌های ورزشی و حتی مجموعه‌های تفریحی و مراکز اداری، متصور بود. کاربر موردنظر تنها با کسب دسترسی لازم و چند کلیک، می‌تواند از وضعیت محیط موردنظر آگاه شده و نسبت به مدیریت آن اقدام کند. لازم به ذکر است که امکان استفاده از هر بخش این سامانه به صورت جعبه سیاه (Black Box) نیز وجود دارد. برای مثال می‌توان صرفاً از بخش پردازش تصویر آن در واحد‌های دیگر بهره گرفت و خروجی RestTemplate آن را دریافت کرده و در اپلیکیشن‌های دیگر نیز مورداستفاده قرارداد.

## 1-3- ساختار پایان نامه

ابتدا در فصل دوم، به بررسی فرایند ایجاد پروژه و ایجاد اتصالات دیتابیس مربوطه می‌پردازیم. چالش‌ها و نکات مربوط به تنظیمات اولیه چنین پروژه‌ای را مرور خواهیم کرد. همچنین فرایند اتصال پروژه به گیت‌هاب از مواردی است که در فصل دوم به آن پرداخته خواهد شد.

پس از بیان مفاهیم راه‌اندازی اولیه، در فصل سوم به سراغ فرایند کدنویسی، تعریف کلاس‌ها و روابط آن‌ها، کامپوننت نویسی و طراحی قالب برنامه، ایجاد صفحات مربوطه، نمایش پیغام‌ها، شیوه بررسی دسترسی‌ها و مدیریت فرایند امنیت سامانه خواهیم رفت و چالش‌های این مسیر مورد بررسی قرار خواهند گرفت.

در فصل چهارم، مباحث مربوط به پردازش تصویر، شیوه‌های پیاده‌سازی، چالش‌ها و راه‌حل‌های آن‌ها مطرح خواهند شد. در فصل نهایی، نتایج، مشکلات و راه‌حل‌های آن‌ها مطرح شده است.

# فصل دوم مفاهیم اولیه

## 2-1- مقدمه

در این فصل، ابتدا به شیوه ایجاد پروژه، تعریف وابستگی‌ها و ایجاد اتصالات دیتابیس مربوطه پرداخته خواهد شد. چالش‌ها و نکات مربوط به تنظیمات اولیه چنین پروژه‌ای را مرور خواهند شد. همچنین مبحث کنترل ورژن از مواردی است که در فصل مورد بررسی قرار می‌گیرد.

## 2-2- ایجاد پروژه

زبان جاوا به همراه فریم‌ورک Spring Boot یک بستر بسیار مناسب برای توسعه برنامه‌های سازمانی و سیستم‌هایی است که قرار است در مقیاس بزرگ و به صورت کاملاً ایمن و پایدار توسعه داده شوند. برای ایجاد پروژه می‌توان به سایت start.spring.io مراجعه نمود. با پر کردن فیلدهای مربوطه، قادر به دریافت پروژه‌ی خام خود خواهیم بود. برای توسعه و کدنویسی پروژه‌های Spring Boot از IDE[[4]](#footnote-5) های مختلفی می‌توان استفاده کرد که معروف‌ترین آن‌ها InteliJ محصول شرکت JetBrains است.

## 2-3- ایجاد وابستگی‌های برنامه

مفهوم وابستگی (Dependency) از مفاهیم پایه در زبان‌های شیءگرا است که از بازنویسی کدها جلوگیری می‌کند. این وابستگی‌ها می‌توانند داخلی و یا خارجی باشند که در این قسمت وابستگی‌های خارجی موردنیاز معرفی شده‌اند. این وابستگی‌ها در فایل pom.xml قابل‌تعریف هستند.

جدول 2-1: وابستگی‌های خارجی موردنیاز برای پروژه

|  |  |
| --- | --- |
| artifactId | GroupId |
| spring-boot-starter-web | org.springframework.boot |
| spring-boot-starter-actuator | org.springframework.boot |
| spring-data-jpa | org.springframework.data |
| spring-boot-starter-data-jpa | org.springframework.boot |
| mysql-connector-java | mysql |
| ojdbc10 | com.oracle.database.jdbc |
| spring-boot-starter-thymeleaf | org.springframework.boot |
| bootstrap | org.webjars |
| jquery | org.webjars.bower |
| spring-boot-starter-tomcat | org.springframework.boot |
| opencv | org.openpnp |

## 2-3- بهره‌گیری از Thymleaf

Thymeleaf یک ابزار مفید برای استفاده از زبان HTML[[5]](#footnote-6) در پروژه‌های Java است که به شما این امکان را می‌دهد که قالب‌های HTML را به صورت پویا در برنامه‌های Spring Framework استفاده کنید. از Thymeleaf معمولاً در تولید و نمایش داده‌ها در صفحات وب استفاده می‌شود.

همان‌طور که در بخش قبل نیز اشاره شد، تعریف وابستگی آن تحت عنوان spring-boot-starter-thymeleaf به فایل pom الزامی است. همچنین برای مهیاشدن شرایط طراحی قالب پروژه، در پوشه‌ی resources دو پوشه‌ی جدید با عنوان‌های static و templates ایجاد شده است.

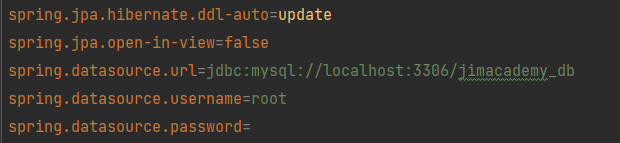
## 2-4- دیتابیس

بسیاری از برنامه‌های نرم‌افزاری نیاز به ذخیره‌سازی و مدیریت داده دارند و دیتابیس‌ها به‌عنوان یک ذخیره‌ساز مرکزی و مؤثر برای این منظور استفاده می‌شوند. در موارد زیادی، دیتابیس به‌عنوان یک منبع اطلاعات مهم برای برنامه‌ها عمل می‌کند و امکانات مختلفی را برای مدیریت و جستجو در داده‌ها ارائه می‌دهد. در این پروژه نیز برای ذخیره‌سازی داده‌های مربوط به کلاس‌ها، دانشجویان، تکالیف و تمرینات، اطلاعات کاربران و همچنین روابط میان این موارد به یک دیتابیس مستقر بر روی سرور احتیاج دارد.

### 2-4-1- MySQL

نرم‌افزار MySQL، یک سیستم مدیریت دیتابیس است که انتخاب خوبی برای توسعه وب‌سایت‌ها یا برنامه‌های کاربردی وب به‌حساب می‌آید. MySQL نرم‌افزاری منبع‌باز است که توسط Oracle پشتیبانی می‌شود.

باتوجه‌به راحتی و مدیریت آسان، در این پروژه نیز از این سیستم مدیریت دیتابیس استفاده شده است که وابستگی آن نیز قبلاً ذکر شده است. برای اتصال پروژه به دیتابیس مذکور، در فایل application.properties نیاز به معرفی آن داریم که به صورت زیر انجام‌گرفته است:



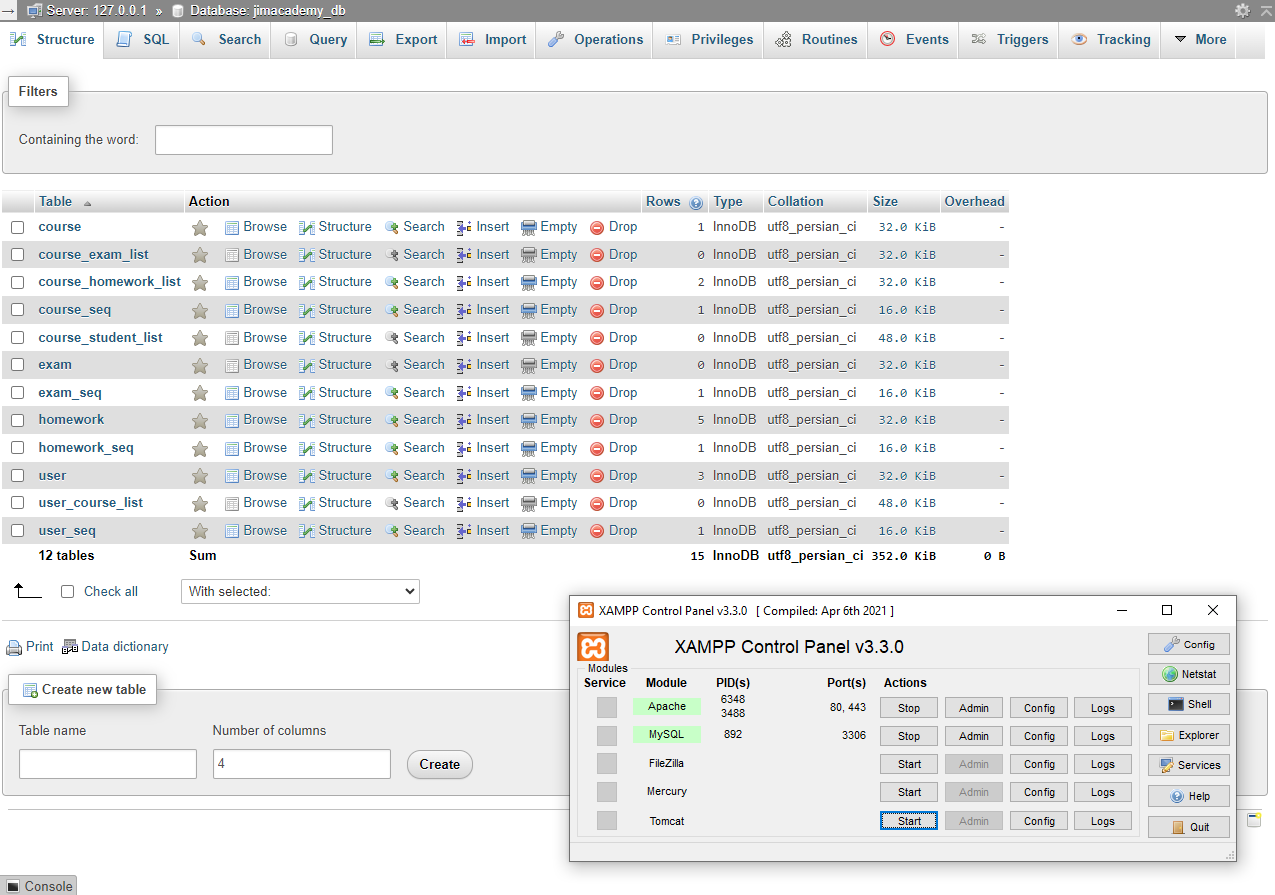
شکل 2-1: نحوه‌ی کانفیگ دیتابیس در APPLICATION.PROPERTIES

لازم به ذکر است که باتوجه‌ به استقرار دیتابیس بر روی لوکال، رمز عبور تعریف نشده است. همچنین در عبارت نخستین، مدیریت جداول دیتابیس، مانند ایجاد، حذف و تغییر نیز به فریم‌ورک سپرده شده است. این مدیریت بر حسب انوتیشین‌های تعریف شده در پروژه صورت می‌گیرد که در فصل بعد به آن‌ها می‌پردازیم.

### 2-4-2- XAMPP

XAMPP یک نرم‌افزاری قابل‌نصب است که برای راه‌اندازی وب سرور محلی (localhost) بر روی سیستم‌های عامل مختلف مانند Windows، Linux، macOS طراحی شده است. اسم XAMPP از اختصار Apache، MySQL، PHP، و Perl گرفته شده است که نشان‌دهنده‌ی اجزای اصلی این برنامه است.

در این پروژه از XAMPP برای اجرای MySQL بهره گرفته شده است.



شکل 2-2: نمای کلی دیتابیس برنامه با استفاده از برنامه XAMPP

## 2-5- کنترل ورژن Git

Git یک سیستم کنترل نسخه توزیع شده است که برای مدیریت کد منبع و تاریخچه تغییرات در پروژه‌های نرم‌افزاری استفاده می‌شود. در Git، هر پروژه دارای یک مخزن (repository) محلی است که حاوی کلیه تاریخچه تغییرات و شاخه‌های مختلف کد است. توسعه‌دهندگان می‌توانند تغییرات خود را اعمال کنند، شاخه‌های جدید ایجاد کنند، با شاخه‌های دیگر ادغام کنند، و تغییرات را به مخزن اصلی master منتقل کنند و پیشرفت پروژه را مشاهده کنند.

در این پروژه، از پلتفرم Github استفاده شده است. همان‌طور که پیش‌تر اشاره شد، این سامانه به صورت دو پروژه‌ی مرتبط پیاده‌سازی شده است و در پلتفرم GitHub نیز این دو پروژه به صورت مجزا تحت عنوان‌های Jim-Academy و JimCalssRoom آپلود شده‌اند که لینک آن‌ها در پیوست ضمیمه شده است.

## 2-6- جمع‌بندي

در این فصل مفاهیم اولیه برای راه‌اندازی فرایند توسعه بیان شدند و مواردی مانند کنترل ورژن، اتصال به دیتابیس و ابزار‌های آن و همچنین تنظیمات برنامه مرور شدند. موارد مذکور از جمله اصول پایه و الزامی برای شروع چنین پروژه‌هایی هستند که می‌توانند به طرق مختلف اجرا شوند.

# فصل سوم شرح کلیات پروژه

## 3-1- مقدمه

پس از مرور مفاهیم در فصل دوم، در این فصل به شیوه‌های پیاده‌سازی این سامانه و معماری آن پرداخته خواهد شد. در این مرحله با مرور تمام کدهای اساسی، کلیات و چارچوب سامانه معرفی خواهند شد. در ادامه‌ی این فصل نیز به کلیات طراحی و پیاده‌سازی ظاهر برنامه اشاره خواهد شد.

## 3-2- کلاس‌ها[[6]](#footnote-7)

در زبان جاوا و دیگر زبان‌های برنامه‌نویسی شی‌ءگرا، کلاس‌ها یکی از مفاهیم اولیه هستند که بستر تعریف متغیر‌ها و توابع محسوب می‌شوند. در این بخش، کلاس‌های اصلی پروژه، کاربرد آنها، روابط و تغییرات آنها مرور خواهند شد.

### 3-2-1- Entity

در فریم‌ورک Spring در زبان جاوا، مفهوم Entity نمایانگر داده‌های ذخیره شده در پایگاه‌داده است. هر Entity معمولاً با یک جدول پایگاه‌داده مرتبط می‌شود و هر ردیف از جدول معادل با یک نمونه Entity خواهد بود. همچنین روابط میان آنها بیانگر روابط میان جداول دیتابیس است. برای تعریف یک Entity در Spring از انوتیشن [[7]](#footnote-8)@Entity استفاده می‌شود. در ادامه هر Entity و کاربرد آن مطرح شده است.

جدول 3-1: هر Entity و کاربرد آن در پروژه

|  |  |
| --- | --- |
| **نام** | **کاربرد** |
| Course | بیانگر یک درس‌آموزشی، شامل لیست دانشجویان، استاد، اطلاعات، لیست امتحانات و لیست تمرینات |
| Exam | بیانگر یک آزمون درسی، شامل درس مربوطه، تاریخ شروع و پایان، اطلاعات و فایل ضمیمه |
| Homework | بیانگر یک تمرین درسی، شامل درس مربوطه، تاریخ شروع و پایان، اطلاعات و فایل ضمیمه |
| User | بیانگر یک کاربر سامانه، شامل نام و نام خانوادگی، اطلاعات، نام کاربری و رمز عبور |
| Student | بیانگر یک دانشجوی سامانه که از کلاس User ارث‌بری کرده است، شامل لیست دروس خود |
| Teacher | بیانگر یک استاد سامانه که از کلاس User ارث‌بری کرده است، شامل لیست دروس خود |

### 3-2-2- Jpa Repositories

JpaRepository یک واسط (Interface) است که در فریم‌ورک Spring Data JPA ارائه شده است. این واسط امکان انجام تراکنش‌های مختلف مربوط به پایگاه‌داده را با Entityها فراهم می‌کند، بدون اینکه نیاز به نوشتن کوئری SQL مستقیم باشد. هر واسط، توابع مفیدی از Entity مورد نظر خود را ارائه می‌دهند. توابعی مانند:

* findById: واکشی رکورد مورد نظر در جدول مربوطه، با استفاده از مولفه‌ی یکتای آن
* : findAll واکشی تمام رکورد‌های موجود در دیتابیس و ارائه‌ی آن‌ها در یک ساختمان لیست
* : save ذخیره‌ی نمونه‌ی مورد‌نظر در دیتابیس

برای استفاده از JpaRepository، ابتدا باید یک رابط (Interface) ساخته شود که از JpaRepository ارث‌بری کرده و Entity موردنظر خود را مشخص کند. در ادامه لیست رابط‌های ساخته‌شده آورده شده است.

* CourseRepository
* ExamRepository
* HomeworkRepository
* StudentRepository
* TeacherRepository
* UserRepository

### 3-2-3- Controllers

در فریم‌ورک اسپرینگ (Spring)، Controller یکی از اجزای اصلی معماری MVC[[8]](#footnote-9) است. Controller در Spring به کلاس‌هایی اطلاق می‌شود که مسئولیت ارتباط بین Model و View را بر عهده دارند. این کلاس‌ها درخواست‌های ورودی از سمت کاربر یا دیگر سرویس‌ها را مدیریت کرده و اطلاعات لازم را برای Model می‌فرستند و سپس با استفاده از یک View مناسب، نتیجه را به کاربر یا سرویس مقصد ارسال می‌کنند.

در توابع Controller، داده‌های دریافتی می‌توانند در قالب ModelAttribute، RequestParam، PathVariable و یا RequestBody به کنترلر ارسال شوند. این پارامترها اطلاعات مختلفی از درخواست ورودی که از سمت کلاینت به سرور ارسال می‌شود را نمایان می‌کنند.

* **ModelAttribute**

از @ModelAttribute برای دریافت داده‌هایی که در فرم‌ها از طریق وب ارسال می‌شوند، استفاده می‌شود. این اطلاعات به صورت مدل یا شیء به کنترلر منتقل می‌شوند.

public String handleFormSubmission(@ModelAttribute UserForm userForm){...

* **RequestParam**

@RequestParam برای دریافت پارامترهایی از URL یا ارسال شده به صورت Query Parameter استفاده می‌شود.

public String handleRequestParameter(@RequestParam String username){…

* **PathVariable**

اگر اطلاعات مورد نیاز در URL به عنوان قسمتی از Path قرار دارند، @PathVariable برای دریافت این اطلاعات استفاده می‌شود.

@GetMapping("/users/{userId}")  
public String handlePathVariable(@PathVariable Long userId) {…

* **RequestBody**

اگر اطلاعات به صورت JSON یا XML از طریق بدنه درخواست ارسال شده باشند، از @RequestBody برای دریافت این اطلاعات به عنوان یک شیء استفاده می‌شود.

public String handleRequestBody(@RequestBody UserData userData) {

در این پروژه‌ نیز، هر View توسط Controller مورد نظر خود نمایش داده‌می‌شود. در ادامه هر Controller ساخته شده در این پروژه و وظایف آن بیان‌ شده است.

جدول 3-2: هر Controller و کاربرد آن در پروژه

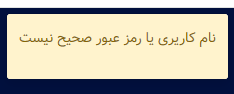
|  |  |  |
| --- | --- | --- |
| **پیشوند URL** | **نام** | **کاربرد** |
| - | MainController | مدیریت صفحات ورود و ثبت‌نام، اعتبارسنجی اولیه و همچنین ایجادکننده‌ی رکورد‌های اولیه در دیتابیس |
| dashboard/ | DashboardController | مدیریت صفحات داشبورد، بخش مدیریت سامانه، ایجاد کلاس جدید و تعریف استاد جدید، بررسی تحرکات کلاس، و همچنین اعتبارسنجی ورود به این صفحات |
| course/ | CourseManagementController | مدیریت بخش کلاس، ایجاد تمرین جدید، ایجاد امتحان جدید و نمایش آن‌ها |
| class/ | ClassRestController | دریافت و پردازش درخواست‌های ارسال شده توسط پروژه ثانویه، مربوط به اعلام مشاهده فعالیت در کلاس |



شکل 3-1: Controller مربوط به داشبورد برنامه

### 3-2-4- Carriers

کلاس‌های Carriers وظیفه‌ی حمل اطلاعات بین دو بخش را به عهده دارند. به عنوان مثال در این پروژه، تنها کلاس Carriers، WebToast نام دارد که اطلاعات لازم برای نمایش پیام‌های Toast را حمل می‌کنند. این اطلاعات شامل نوع پیام (پیام تأیید، پیام اطلاعات، پیام هشدار و پیام خطا)، عنوان پیام و متن پیام است.



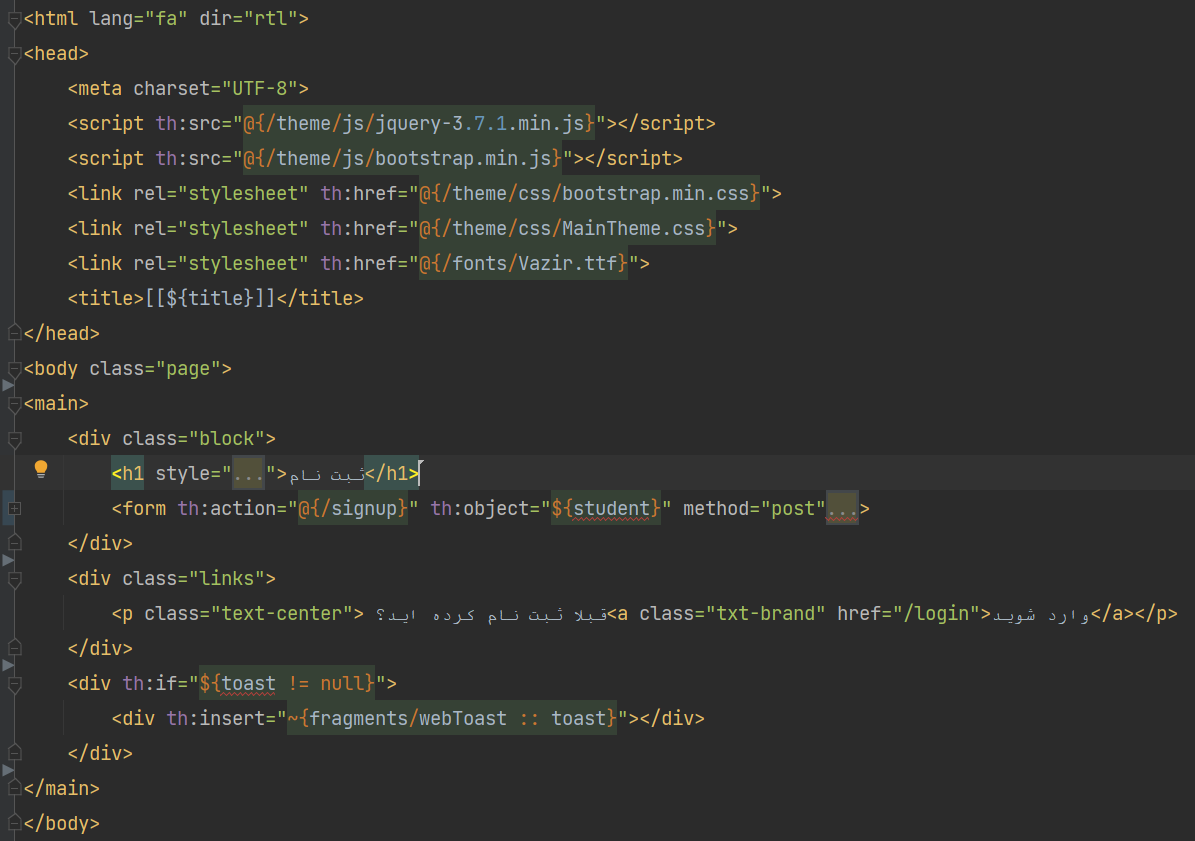
شکل 3-2: یک نمونه پیغام Toast نمایش داده شده در سامانه

## 3-3- طراحی

در این برنامه‌ها از ابزار Thymleaf برای ارتباط با بخش ظاهری (front-end) بهره گرفته شده است. Thymeleaf یک موتور قدرتمند و یک ابزار مفید است که به صورت ویژه برای توسعه وب در چارچوب فریم‌ورک Spring طراحی شده است. این ابزار امکان ادغام آسان و انعطاف‌پذیری با HTML را فراهم می‌کند و به توسعه‌دهندگان امکان ایجاد و نمایش قالب‌های داینامیک در برنامه‌های وب را می‌دهد. Thymeleaf به دلیل سادگی استفاده و امکانات زیادی که ارائه می‌دهد، در توسعه برنامه‌های وب با Spring بسیار محبوب است و به عنوان یکی از ابزارهای محبوب قالب‌گذاری در جهان جاوا شناخته می‌شود.

همچنین بخش ظاهری پروژه با استفاده از زبان‌های HTML[[9]](#footnote-10)، CSS[[10]](#footnote-11)، JavaScript و فریم‌ورک Bootstrap بهره گرفته شده است. Bootstrap یک فریم‌‍ورک طراحی و توسعه وب متن‌باز است که توسط شرکت Twitter ایجاد شده است. این چارچوب حاوی مجموعه‌ای از قالب‌ها، کامپوننت‌ها، و ابزار‌های JavaScript آماده است.

همچنین اکثر صفحات به کمک کتابخانه‌ی jQuery نوشته شده است. jQuery یک کتابخانه‌ی JavaScript است که به تسهیل برنامه‌نویسی در محیط مرورگرها مخصوصاً برای عملیاتDOM[[11]](#footnote-12) و ایجاد افکت‌ها و ارتباط با سرور به صورتAJAX[[12]](#footnote-13) ، مورد استفاده قرار می‌گیرد. در ادامه به کلیات طراحی برنامه اشاره خواهد شد.



شکل 3-3: صفحه‌ی ثبت‌نام برنامه، طراحی شده با Thymeleaf

### 3-3-1- ساختار

تمام فایل‌های مربوط به طراحی و اجرا، درون پوشه‌ی resources قرار گرفته‌اند. این پوشه به دو قسمت static و tempalate تقسیم می‌شود. در بخش static، تمام فایل‌های ثابت و موردنیاز در تمام پروژه واقع شده‌اند. این پوشه شامل فونت‌ها، تصاویر، کتابخانه‌های مربوط به‌ظاهر برنامه و تم برنامه است.

در پوشه‌ی دیگر با نام template، فایل HTML صفحات برنامه و فرگمنت‌ها[[13]](#footnote-14) حضور دارند. فرگمنت‌ها قطعه‌هایی هستند که در بخش‌های مختلف برنامه import شده‌اند [1].



شکل 3-4: پوشه‌بندی بخش ظاهری برنامه

## 3-4- اعتبارسنجی

در این پروژه، بحث امنیت و سطح دسترسی بسیار حائز اهمیت است. ازاین‌رو لازم است تا دسترسی افراد محدود شده و هر کاربر بتواند تنها بخش‌هایی که به آن دسترسی دارد را مشاهده کند. این استراتژی در تابعی با عنوان returnUserValidationModel پیاده‌سازی شده است. به صورتی که با بررسی اطلاعات کاربری که وارد شده است و تطابق آن، ادمین یا اپراتور بودن کاربر تمیز داده می‌شود و همچنین درصورتی‌که اطلاعات کاربر صحیح نباشد و تطابقی یافت نشود، کاربر به صفحه ورود هدایت شود.

همانطور که مشاهده می‌شود، در این تابع از مفهوم Session برای نگهداری داده‌های کاربر وارد شده استفاده شده است. در فریم‌ورک اسپرینگ (Spring)، Session به مجموعه‌ای از داده‌های مربوط به یک کاربر خاص در طول یک جلسه (Session) وب اشاره دارد. جلسه‌ (Session) در دنیای وب برای نگهداری اطلاعاتی همچون وضعیت ورود کاربر، داده‌های فرم، و سایر اطلاعات مربوط به جلسه کاربری استفاده می‌شود. در اسپرینگ، مدیریت Session به وسیله‌ی یک کلاس به نام HttpSession انجام می‌شود. این کلاس امکان ذخیره سازی و بازیابی اطلاعات مختلف را ارائه می‌دهد. داده‌های هر جلسه (Session) پس از مدت مشخصی منقضی می‌شوند.



شکل 3-5: ساختار تابع اعتبار سنجی

## 3-5- جمع‌بندي

در این فصل معماری برنامه به طور کلی مرور و بررسی شد تا در کنار مفاهیم بیان شده در فصل دوم، دید کاملی درباره ساختار و عملکرد کلی پروژه تشکیل شود. همچنین شیوه‌ی اجرای این معماری در دو بخش Front-end و Back-end مورد بررسی قرار گرفت. در انتهای این بخش نیز به نحوه‌ی اعتبارسنجی کاربران و تعیین دسترسی آنان پرداخته شد.

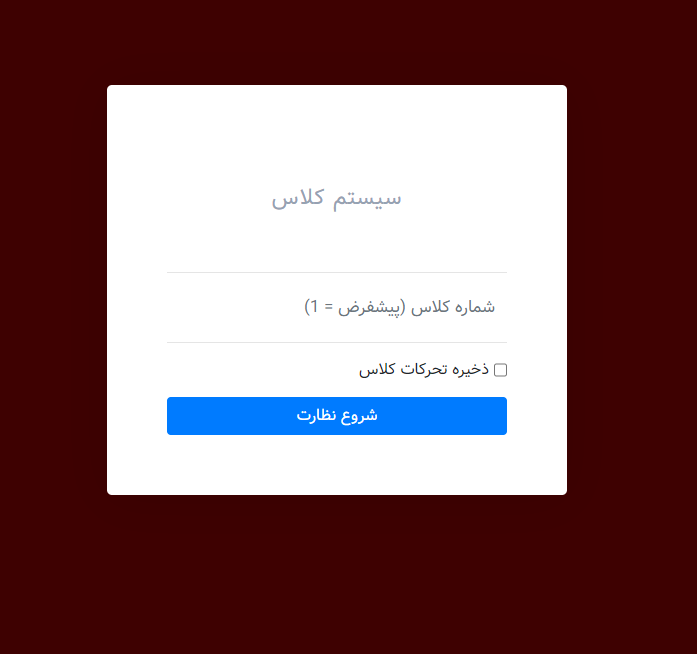
# فصل چهارم پردازش تصویر

## 4-1- مقدمه

در فصل‌های قبل ساختار و رفتار برنامه معرفی شدند. در این فصل، به بخش ویژه‌ی آن یعنی پردازش تصاویر دریافتی از کلاس پرداخته خواهد شد. همان‌طور که گفته شد هدف اصلی و نوین این پروژه، تشخیص حضور افراد در هر کلاس و یا آزمایشگاه است که به صورت یک سیستم بینایی ماشین بلادرنگ پیاده‌سازی شده است. این سامانه همچنین امکان ذخیره‌ی تصویر افراد حاضر در کلاس را نیز میسر می‌کند.

## 4-2- معرفی کاربری

به‌طور کلی ساختار کلی این بخش بدین صورت است که صفحه مخصوص هر کلاس یا آزمایشگاه بر روی مرورگر آن نمایش داده می‌شود. اپراتور پس از تعیین شماره کلاس موردنظر و همچنین تعیین ذخیره‌شدن و یا عدم ذخیره تصویر تحرکات کلاس، بر روی دکمه‎ی «شروع نظارت» کلیک می‌کند. همچنین هر زمان که قصد توقف آن را داشته باشد، بر روی «توقف نظارت» کلیک می‌کند.



شکل 4-1: صفحه مخصوص کلاس، قبل از شروع نظارت

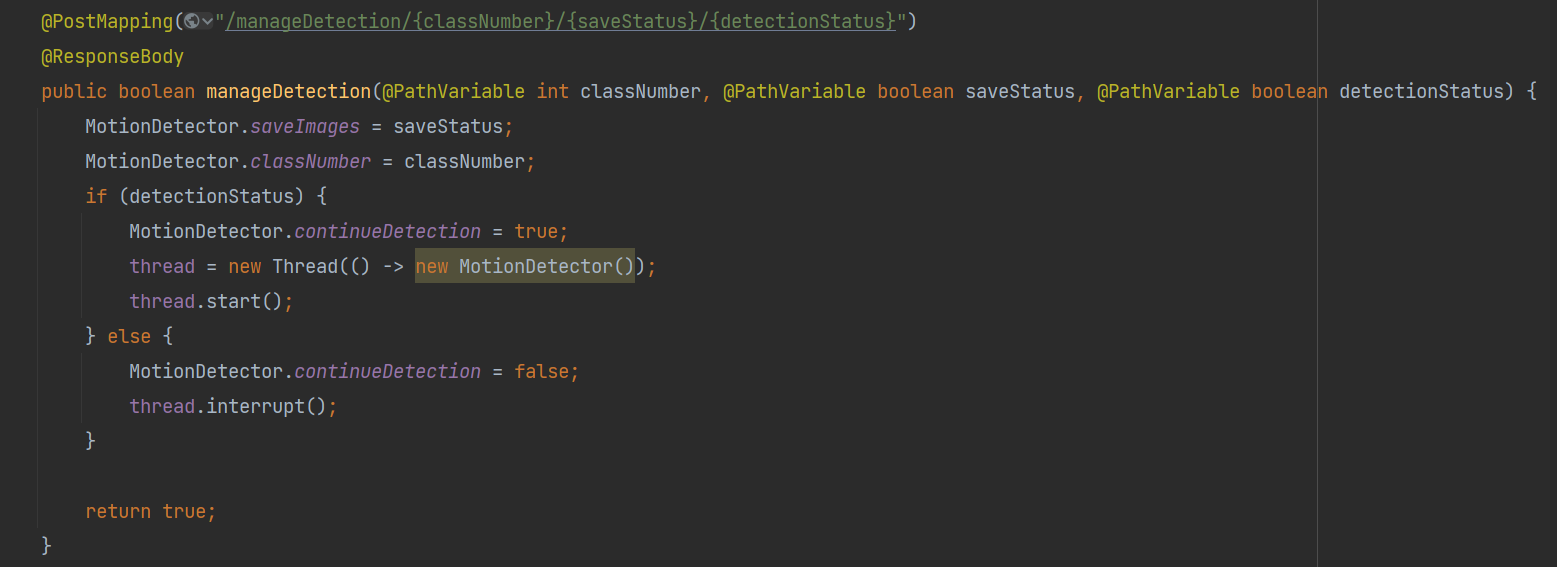
## 4-3- معرفی عملکرد برنامه

قبل از شروع برنامه، کتابخانه‌ی OpenCV که پیش‌تر نیز معرفی شد، واکشی می‌شود. در زمانی که دستور شروع نظارت در برنامه صادر می‌شود، برنامه با ذخیره‌کردن شماره کلاس و همچنین تنظیم کاربر مبنی بر ذخیره و یا عدم ذخیره تصاویر، شروع به کار می‌کند. این فرایند در ادامه به صورت کامل شرح داده شده است.

### 4-3-1- شروع و توقف نظارت

پس از دریافت فرمان شروع نظارت و تنظیم‌شدن مقادیر لازم، برنامه یک Thread جداگانه را ایجاد کرده و یک نمونه از کلاس MotionDetector در آن ایجاد می‌کند. دلیل استفاده از یک Thread جداگانه این است که فرایند برداشت فریم و توقف‌های مکرر مزاحمتی برای Thread اصلی اجرای برنامه ایجاد نکند.

همچنین در زمان صدور فرمان توقف، یک فلگ تعیین‌کننده برای توقف برنامه برابر با مقدار false می‌شود و همچنین متد interrupt در Thread ایجاد شده فراخوانی می‌شود تا فرایند نظارت خاتمه پیدا کند.



شکل 4-2: فرایند شروع و پایان نظارت

### 4-3-2- تابع سازنده‌ی کلاس MotionDetector

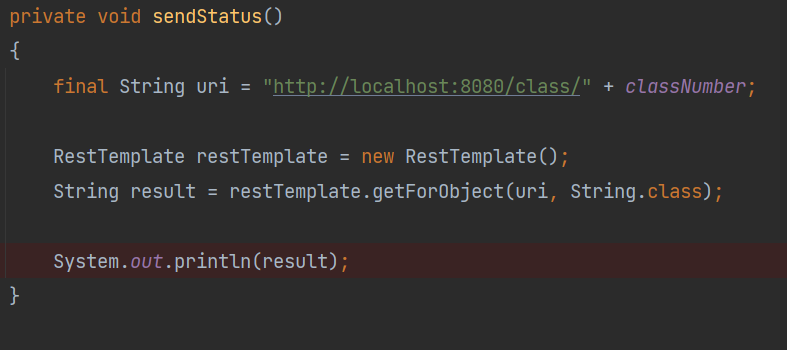
سازنده (Cunstructor) این کلاس به‌محض ایجاد، دوربین سیستم را بازکرده و مادامی که دستور توقف صادر نشده باشد، تابع startDetecting را در یک حلقه فراخوانی می‌کند. پس از صدور فرمان توقف، دوربین آزاد شده و تابع سازنده پایان می‌یابد.

### 4-3-3- تابع startDetecting

پردازش اصلی این بخش در این تابع شکل می‌گیرد. این تابع هر ثانیه، یک فریم از دوربین را برداشت می‌کند. این فریم را به ابعاد کوچک‌تر تقلیل داده، ترکیب رنگی آن را به سیاه‌وسفید تغییر داده و کمی آن را بلور می‌کند. دلیل این فرایند، حذف نویزهای احتمالی بر اثر کیفیت پایین دوربین‌های موجود است. این تغییرات به واسطه‌ی کلاس Imgproc از کتابخانه‌ی OpenCV انجام می‌پذیرد. در نهایت این فریم با فریم قبلی مقایسه شده و در صورت مشاهده تفاوت میان این دو فریم، تابع sendStatus فراخوانی می‌شود و در صورت تنظیم کاربر، تصویر ذخیره می‌شود. در غیر این صورت این فرایند ثانیه‌ی بعد تکرار خواهد شد.

### 4-3-4- تابع sendStatus

این تابع، وظیفه‌ی ارسال وضعیت به سرور را عهده‌دار است. این ارسال در قالب RestTemplate صورت‌گرفته و محتوای آن صرفاً شامل شماره کلاس می‌شود.



شکل 4-3: محتویات تابع ارسال کننده RestTemplate

### 4-3-5- تابع setStatus

این تابع بر روی سرور حضور دارد و وظیفه‌ی دریافت درخواست‌های ارسال شده در هر کلاس را عهده‌دار است. هر دریافت درخواست، به منزله‌ی مشاهده‌ی تحرک در یکی از کلاس‌هاست. این تابع با هر دریافت، شماره کلاس و زمان دقیق دریافت دیتا را در یک ساختمان داده‌ی Map اضافه می‌کند. کلید این Map همان شماره کلاس است و بدین ترتیب به‌ازای هر کلاس فقط یک رکورد (آخرین زمان دریافت دیتا) وجود دارد. در زمان نمایش اطلاعات، هر کلاسی که آخرین تحرک دریافتی از آن مربوط به کمتر از 3 ثانیه‌ی قبل باشد به معنی کلاس فعال تلقی می‌گردد.



شکل 4-4: تابع تمیز دهنده کلاس‌های فعال و غیرفعال



شکل 4-5: شیوه‌ی نمایش کلاس‌های فعال و غیر‌فعال در سامانه

## 4-4- جمع‌بندي

در این فصل، نحوه‌ی کار با بخش پردازش تصویر آموزش داده شد و سپس عملکرد سیستم در پس‌زمینه بررسی شد. در ادامه ساختار کلی واحد پردازش تصویر مطرح شد. در نهایت شیوه‌ی انتقال دستور شروع و توقف، استخراج فریم‌ها، شیوه‌ی مقایسه و تشخیص، نحوه‌ی ارتباط ماژول‌ها با یکدیگر توسط RestTemplate و همچنین نحوه‌ی پیاده‌سازی تمام این موارد شرح داده شدند.

# فصل پنجم چالش‌ها و نتیجه‌گیری

## 5-1- چالش‌ها و راه‌حل‌های آن‌ها

در مسیر اجرای این پروژه، چالش‌ها و مشکلات متعددی حضور داشتند. کمبود تجربه در عرصه‌ی پردازش تصویر و همچنین انتخاب زبانی به گستردگی جاوا برای یک پروژه‌ی دانشگاهی، از مواردی بودند که این مسیر را بیش ‌از پیش دشوار کردند. اما با وجود تمام این مشکلات و حتی ضیق وقت، عطش سیری‌ناپذیر برای یادگیری و کسب تجربیات جدید نیروی محرک روبه‌جلو برای اتمام این پروژه بوده است. در ادامه برخی از چالش‌های فنی موجود در مسیر، مورد بررسی قرار گرفته‌اند.

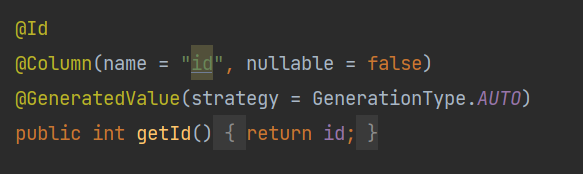
### 5-1-1- Id generation

در هنگامی Entityها تعریف می‌شدند، فیلد Id نیز در هر Entity ایجاد می‌شد. اما این فیلد توسط کاربر پر نمی‌شد و مقدار آن الزاماً باید یکتا باشد. در تجربیات قبلی، این چالش با تعریف Sequence در دیتابیس مرتفع می‌شد، اما دیتابیس مورداستفاده در این پروژه در حالت اولیه دارای Sequence نبود و باتوجه‌ به reset شدن مداوم آن، چنین تغییری در هر اجرا منطقی به نظر نمی‌رسید، علاوه‌بر آن تصمیم بر آن بود که تمام مدیریت دیتابیس به فریم‌ورک سپرده شود.

پس از آزمایش‌های مکرر و جستجوی راه‌حل، در نهایت استفاده از انوتیشن زیر بهینه‌ترین راه‌حل بود:

@GeneratedValue(strategy = GenerationType.AUTO)

اما این راه‌حل با دیتا تایپ Long پاسخگو نبود. پس از آزمون و خطای بسیار، این مورد با تغییر این دیتا تایپ به int مرتفع گردید [2].



شکل 5-1: نمونه استفاده‌ی @GeneratedValue در پروژه

### 5-1-2- جابجایی میان توابع Controller به همراه مقادیر

در صفحاتی مانند صفحه‌ی ورود، لازم بود تا در یکی از توابع پردازشی انجام شود و برای مثال در صورت تأیید اطلاعات وارد شده، کاربر به صفحه‌ی دیگری منتقل شود. راه‌حل اولی که به ذهن می‌رسد، ریدایرکت کردن کاربر است [4]. اما در این سناریو لازم بود علاوه بر انتقال کاربر، اطلاعات دیگری نیز به تابع مقصد ارسال شوند.

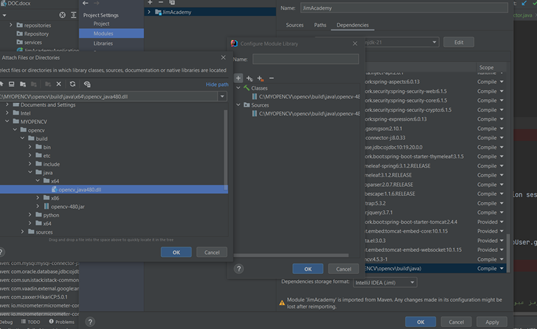
پس از تحقیق و جستجو، استفاده از FlashAttribute بهترین راه‌حل بود. این ویژگی می‌بایست به همراه RedirectView منتقل می‌شد. این چالش نیز پس از آزمایش بسیار مرتفع شد.



شکل 5-2: یک نمونه استفاده‌ی RedirectView در پروژه

### 5-1-3- OpenCV

باتوجه به نبود هیچ‌گونه تجربه در زمینه‌ی پردازش تصویر، شروع فعالیت در این زمینه یکی از اصلی‌ترین چالش‌ها بود. شروع این فرایند نیازمند تحقیق بسیار بود. باتوجه به این کمبود تجربه، حتی هیچ‌گونه دیدگاهی نسبت شیوه‌ی پیاده‌سازی وجود نداشت. نمونه‌های موجود نیز به زبان جاوا نیستند. با امتحان شیوه‌های مختلف، تصمیم بر آن شد که از کتابخانه‌ی OpenCV استفاده شود. اما نسبت به استفاده از کتابخانه‌های خارجی نیز هیچ‌گونه تصوری وجود نداشت. در نهایت با مطالعه‌ی آموزش‌های مختلف، با تعریف این کتابخانه در ProjectSetting و همچنین تعریف فایل dll به صورت دستی [3] OpenCV نصب شد و اتصال آن به پروژه نیز با موفقیت انجام شد.



**شکل 3-5: فرایند تعریف کتابخانه‌ی OpenCV**

## 5-2- نتیجه‎گیری

در این فصل برخی از مشکلات و چالش‌های پیش‌آمده در مسیر این پروژه مطرح شده‌اند که هرکدام با تحقیق و آزمون به‌خوبی برطرف شده‌اند. این پروژه به همراه تمام چالش‌های آن، سکوی پرتابی بوده است برای یادگیری مفاهیم جدید، تجربه‌ی بیشتر و رشد فکری هرچه بیشتر.

## پيوست 1: لینک گیت‌هاب پروژه‌ها

* <https://github.com/MahdiJafari26/Jim-Academy>
* <https://github.com/MahdiJafari26/Jim-Academy>

## منابع

[1] P. Mihandoost, "A very quick start – Thymeleaf," Virgool, January 2024, https://vrgl.ir/nvRF8.

[2] D. Pigott, "An Overview of Identifiers in Hibernate/JPA," Baeldung, January 2024, https://www.baeldung.com/hibernate-identifiers.

[3] BS tech, "How to Set up OpenCV for Java in Intellij Idea," Youtube, January 2024, youtu.be/TsUhEuySano?si=7-pT2Zn0lHkAyyaU.

[4] J. S. Patel, M. K. Smith, S. W. Wang, "Spring in Action: Building Robust Enterprise Applications," TechBooks International, 2022.

1. HyperText Markup Language [↑](#footnote-ref-2)
2. Cascading Style Sheets [↑](#footnote-ref-3)
3. Open computer vision [↑](#footnote-ref-4)
4. Integrated Development Environment [↑](#footnote-ref-5)
5. HyperText Markup Language [↑](#footnote-ref-6)
6. Classes [↑](#footnote-ref-7)
7. Annotation [↑](#footnote-ref-8)
8. Model-View-Controller [↑](#footnote-ref-9)
9. HyperText Markup Language [↑](#footnote-ref-10)
10. Cascading Style Sheets [↑](#footnote-ref-11)
11. Document Object Model [↑](#footnote-ref-12)
12. Asynchronous JavaScript and XML [↑](#footnote-ref-13)
13. Fragments [↑](#footnote-ref-14)