



دانشگاه اصفهان

دانشکده مهندسی کامپیوتر

گروه مهندسی نرم افزار

گزارش پروژه کارشناسی

توسعه‌ی بستر یکپارچه بلاگ و فروم با قابلیت انعطاف پذیری بالا

**استاد راهنما:**

دکتر بهروز ترک لادانی

**پژوهشگر:**

مهدی شیری

بهار ۱۳۹۹

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

## چکیده

امروزه که به عصر ارتباطات نیز مشهور شده است، انسان‌ها از راه‌های مختلفی به غیر از دید و بازدید ساده با هم به تبادل اطلاعات می‌پردازند و دیگر افراد محدود به شهر و یا حتی کشور خود نیستند. در میان این راه‌ها، بخش عمده‌ی ارتباطات افراد در بستر اینترنت یا به اصطلاح فضای مجازی است.

در این پروژه با در نظر گرفتن این نیاز جامعه و یکپارچه‌سازی نیازهای مختلف ارتباطی در یک بستر واحد و با تاکید بر احترام به سلیقه‌ی افراد و نیازمندی‌های متفاوت آن‌ها، سعی شده است، وب اپلیکیشن مفید و کاربردی ارائه شود تا ساخت و استفاده اجتماعی‌هایی نظیر بلاگ و انجمن در زمانی کم و با هزینه‌ای بسیار پایین میسر شود.

این وب اپلیکیشن کاربردی مطابق با اصول مهندسی نرم‌افزار با استفاده از چارچوب انگولار برای بخش فرانت‌اند، چارچوب نست‌جی‌اس برای بخش بک‌اند و با بهره‌گیری از دیتابیس مونگو پیاده‌سازی شده و در توسعه آن از برنامه‌نویسی شی‌گرا و اصول طراحی واسط استفاده شده است. برنامه‌ی کاربردی نوشته شده و مستندات مربوط به آن که حاصل کار پروژه است، پایه‌ای برای نسخه‌های ارتقا یافته‌ی بعدی که قابل گسترش و تجاری‌سازی هستند خواهد بود.

**واژگان کلیدی:** ارتباطات مجازی، انعطاف‌پذیری، انگولار، چارچوب، مونگو، نست‌جی‌اس، واسط کاربری

## فهرست موضوعی

عنوان	صفحه
<b>فصل اول: کلیات</b>	<b>۱</b>
۱-۱- مقدمه	۱
۲-۱- نیازسنجی	۱
۳-۱- روش و هدف انجام پروژه	۲
۴-۱- بررسی نمونه‌های تجاری مشابه	۳
۵-۱- ساختار پایان‌نامه	۳
<b>فصل دوم: مفاهیم</b>	<b>۵</b>
۱-۲- مقدمه	۵
۲-۲- زبان برنامه‌نویسی تایپاسکریپت	۵
۳-۲- چارچوب‌های نرم‌افزاری	۶
۱-۳-۲- چارچوب انگولار	۶
۲-۳-۲- چارچوب نست‌جی‌اس	۸
۴-۲- طراحی متریکال برای رابط کاربری/تعاملی	۸
۵-۲- معماری و ساختار برنامه‌نویسی	۹
۶-۲- متدولوژی توسعه افزایشی و تکراری	۱۱
۷-۲- جمع بندی	۱۱
<b>فصل سوم: تحلیل و مستندسازی نرم‌افزار</b>	<b>۱۲</b>
۱-۳- مقدمه	۱۲
۲-۳- سند نیازمندی‌های نرم‌افزار (SRS)	۱۲
۱-۲-۳- هدف	۱۳
۲-۲-۳- قلمرو	۱۳
۳-۲-۳- شرح کلی	۱۳

۱۳	۳-۲-۴- کارکرد محصول
۱۴	۳-۲-۵- مشخصات کاربر
۱۴	۳-۲-۶- قیود
۱۴	۳-۲-۷- مفروضات و نیازمندی‌ها
۱۴	۳-۲-۸- نیازمندی‌ها
۱۷	۳-۳- استنتاج مورد کاربردها از نیازمندی‌ها
۱۷	۳-۳-۱- شناسایی مورد کاربردها
۲۶	۳-۳-۲- مصورسازی مورد کاربردها
۲۷	۳-۴- مدل‌سازی تعامل کنش‌گر - سیستم
۲۸	۳-۴-۱- استفاده از پیش‌نمونه‌های واسط کاربری
۳۱	۳-۵- مدل‌سازی تعامل شیء
۳۱	۳-۵-۱- شناسایی گام‌های غیر بدیهی
۳۳	۳-۵-۲- نوشتن سناریو برای گام غیربدیهی
۳۳	۳-۵-۳- استنتاج نمودارهای توالی از سناریو
۳۴	۳-۶- استنتاج نمودار کلاس طراحی
۳۵	۳-۶-۱- به تصویر کشیدن نمودار کلاس طراحی
۳۵	۳-۷- جمع‌بندی
۳۶	<b>فصل چهارم: پیاده‌سازی</b>
۳۶	۴-۱- مقدمه
۳۶	۴-۲- چارچوب انگولار
۳۶	۴-۲-۱- استفاده از چارچوب انگولار
۳۸	۴-۲-۲- ساختار پروژه‌ها در انگولار
۳۹	۴-۲-۳- اتصال داده در انگولار
۴۰	۴-۲-۴- Lazy Loading در انگولار
۴۱	۴-۲-۵- مدیریت وضعیت

۴۳	۴-۲-۶- محافظ‌ها در انگولار
۴۳	۴-۳- چارچوب نست‌جی‌اس
۴۳	۴-۳-۱- استفاده از چارچوب نست‌جی‌اس
۴۴	۴-۳-۲- ساختار پروژه در نست‌جی‌اس
۴۵	۴-۴- Meta-Programming در نست‌جی‌اس
۴۵	۴-۵- جمع‌بندی
۴۶	فصل پنجم: تست و آزمون نرم‌افزار
۴۶	۵-۱- مقدمه
۴۷	۵-۲- تست واحد
۴۸	۵-۲-۱- تست واحد در فرانت‌اند
۵۲	۵-۲-۲- تست واحد در بخش بک‌اند
۵۳	۵-۳- آزمون یکپارچه‌سازی
۵۴	۵-۴- آزمون رابط کاربری یا E2E
۵۴	۵-۵- آزمون واسط کاربری
۵۵	۵-۶- جمع‌بندی
۵۹	فصل ششم: جمع‌بندی و پیشنهادها
۵۹	۶-۱- جمع‌بندی و دورنما
۶۰	۶-۲- پیشنهادها
۶۱	منابع
۶۲	پیوست یک (کتابخانه‌های استفاده شده در فرانت‌اند)
۶۳	پیوست دو (افزونه‌های استفاده شده در VS Code)
۶۴	پیوست سه (کتابخانه‌های استفاده شده در بک‌اند)

## فهرست شکل‌ها

عنوان	صفحه
شکل ۱: معماری کلی انگولار [۱].....	۷
شکل ۲: انگولار و نست‌جی‌اس هر دو از زبان تایپ‌اسکریپت استفاده می‌کنند.....	۸
شکل ۳: استفاده از مولفه‌ها در دیگر مولفه‌ها.....	۹
شکل ۴: نمونه‌ای از لوله در انگولار.....	۱۰
شکل ۵: نمودار مورد کاربردها.....	۲۶
شکل ۶: صفحه اصلی انجمن.....	۲۸
شکل ۷: صفحه اصلی بلاگ.....	۲۸
شکل ۸: تنظیمات انجمن - اطلاعات کلی (قسمت یک).....	۲۹
شکل ۹: تنظیمات انجمن - اطلاعات کلی (قسمت دو).....	۲۹
شکل ۱۰: تنظیمات وبلاگ - اطلاعات کلی.....	۳۰
شکل ۱۱: تنظیمات ویجت‌ها.....	۳۰
شکل ۱۲: تنظیمات انجمن - مجوزها و دسترسی.....	۳۱
شکل ۱۳: سناریو گام غیر بدیهی مورد کاربر ۱۸ (تغییر مشخصات کاربری).....	۳۳
شکل ۱۴: نمودار توالی مورد کاربرد ۱۸ (تغییر مشخصات کاربری).....	۳۴
شکل ۱۵: نمودار کلاس.....	۳۵
شکل ۱۶: اولین پروژه‌ی انگولار.....	۳۷
شکل ۱۷: ساختار فایل در انگولار.....	۳۸
شکل ۱۸: اضافه کلاس CSS به صورت شرطی به یک تگ.....	۴۰
شکل ۱۹: چرخه‌ی حیات معماری Redux در انگولار (کتابخانه NgRx).....	۴۲
شکل ۲۰: محافظ در مسیریابی.....	۴۳
شکل ۲۱: ساختار پروژه در نست‌جی‌اس.....	۴۴
شکل ۲۲: استفاده دکوراتورها و پایپ در قسمتی از یک کنترلر.....	۴۵
شکل ۲۳: سطوح تست نرم‌افزار در پروژه.....	۴۷
شکل ۲۴: آزمون واحد برای مولفه app.....	۴۸
شکل ۲۵: آزمون واحد برای محافظ Auth.....	۴۹
شکل ۲۶: آزمون لوله تبدیل تاریخ شمسی.....	۵۰
شکل ۲۷: آزمون واحد Social Effects به طور خلاصه.....	۵۱
شکل ۲۸: آزمون واحد Social Reducer.....	۵۱

- شکل ۲۹: آزمون کنترلر search ..... ۵۲
- شکل ۳۰: آزمون یکپارچه‌سازی انجام لاگین در سایت ..... ۵۳
- شکل ۳۱: نتیجه آزمون E2E انجام شده که همه‌ی آن با موفقیت گذرانده شده است. .... ۵۴
- شکل ۳۲: نظرسنجی واسط کاربری (رضایت‌مندی) ..... ۵۵
- شکل ۳۳: نظرسنجی واسط کاربری (قابلیت یادگیری) ..... ۵۶
- شکل ۳۴: نظرسنجی واسط کاربری (قابلیت رویت) ..... ۵۶
- شکل ۳۵: نظرسنجی واسط کاربری (کارایی) ..... ۵۷
- شکل ۳۶: نظرسنجی واسط کاربری (کنترل خطا و کاربر) ..... ۵۷



## فصل اول

### کلیات

#### ۱-۱ - مقدمه

امروزه اینترنت عضو جدایی ناپذیر زندگی انسان شده است و سهم عظیمی از وقت هر فرد در این محیط صرف می شود. پیشرفت روزافزون چنین فضایی باعث شده است که تعاملات انسانی شکل های جدیدتری به خود بگیرد و بسیاری از ارتباطات که قبلاً ناممکن به نظر می رسید یا به شکل حضوری صورت می پذیرفت به شکل غیرحضوری و به اصطلاح مجازی ممکن شود. به همین دلیل، فرصت بسیار خوبی است تا با شکل گیری محیطی مناسب، فضایی برای تعامل و مطالعه ی افراد در مورد موضوعاتی که به آن علاقه مند هستند به خصوص برای وب فارسی فراهم شود.

#### ۱-۲ - نیازسنجی

به نظر می رسد در وب فارسی، بستری مناسب برای تعامل افراد در مورد موضوعات مختلف و البته با دسته بندی مناسب و قابلیت جستجوی بالا از موتورهای جستجو<sup>۱</sup> وجود ندارد. همچنین پیچیدگی راه اندازی و هزینه های آن باعث می شود تا افراد علاقه مند، رغبتی به ساخت فضای جدید در مورد موضوعی خاص پیدا نکنند و این اتفاق باعث می شود تا اطلاعات بسیار ارزنده که طی مدت زمانی قابل توجه توسط افراد جمع آوری

---

<sup>1</sup> Search Engine

شده‌اند در وب عمیق<sup>۱</sup> مانند اپلیکیشن‌های پیام‌های فوری<sup>۲</sup> و یا در ذهن افراد دفن شوند [۱].

### ۱-۳- روش و هدف انجام پروژه

با توجه به تمایل مردم به فضای مجازی چه برای سرگرمی و چه برای گسترش کسب‌وکار و یا حتی ماهیت آن باید محیطی جامع بنا نهاد. البته باید به موضوع بسیار مهمی هم توجه داشت که بسیاری از این افراد شناخت کمی از تکنولوژی دارند. پس باید علاوه بر برآورده‌سازی جامعیت نیازهای اجتماعی محیطی با رابط کاربری ساده و با امنیت بالا را برای کاربران به ارمغان آورد. همچنین به دلیل رعایت استانداردها و اصول مهندسی نرم‌افزار و استفاده از متدولوژی<sup>۳</sup> توسعه افزایشی<sup>۴</sup>، می‌توان این استارت‌آپ<sup>۵</sup> را با توجه به نیازهای جامعه کامل‌تر نمود تا این محصول با نیازهای جامعه وفق پیدا کند و همراه آنان باشد.

در توسعه این وب‌اپلیکیشن سعی شده است از ابزارها و تکنولوژی‌های استفاده شود که قابلیت نگهداری بالا داشته باشند تا هر قدمی که در متدولوژی بیان شده برداشته می‌شود هزینه کمتری پرداخته شود.

تکنولوژی استفاده شده در بخش سرور Nginx است که علاوه بر سرعت بسیار بالا شامل ابزارهای بسیار زیادی مانند «کش‌کننده محتوا»<sup>۶</sup>، «پراکسی معکوس»<sup>۷</sup>، «تعادل بار»<sup>۸</sup> و کنترل‌کننده فایل‌های ایستا<sup>۹</sup> اشاره نمود.

در بخش بک‌اند<sup>۱۰</sup> چارچوب<sup>۱۱</sup> نست‌جی‌اس<sup>۱۲</sup> که در واقع پوششی<sup>۱۳</sup> بر روی چارچوب‌های مطرحی چون اکسپرس<sup>۱۴</sup> و فستیفای<sup>۱۵</sup> است، مورد استفاده قرار گرفته است. با استفاده از این تکنولوژی برخلاف چارچوب‌های لایه‌ی زیرین آن، معماری و استاندارد یک‌ه‌ای تعبیه شده است و زبان پیش‌فرض آن از جاوا اسکریپت به تایپ‌اسکریپت که زبانی مشابه جاوا اسکریپت ولی تایپ قوی<sup>۱۶</sup> است بهره گرفته شده است.

---

<sup>1</sup> Deep Web

<sup>2</sup> Instant Messaging

<sup>3</sup> Methodology

<sup>4</sup> Incremental Development

<sup>5</sup> Startup

<sup>6</sup> Content Caching

<sup>7</sup> Reverse Proxy

<sup>8</sup> Load Balancing

<sup>9</sup> Static Files

<sup>10</sup> Back-End

<sup>11</sup> Framework

<sup>12</sup> NestJS

<sup>13</sup> Wrapper

<sup>14</sup> Express

<sup>15</sup> Fastify

<sup>16</sup> Strongly Typed

برای پایگاه داده از تکنولوژی NoSQL سندگان با نام مونگو<sup>۱</sup> استفاده شده است. هدف استفاده از چنین ابزاری به جای جایگزین های ساختاریافته ی آن (SQL)، پرداخت هزینه ی حافظه و نوشتن چندباره<sup>۲</sup> در بعضی موارد در مقابل مصرف کمتر پردازش گر و سرعت بیشتر خواندن خواهد بود. همچنین دلیل دیگر قابلیت گسترش پذیری راحت تر نسبت به رقبای دیگر آن است.

برای فرانت اند<sup>۳</sup> نیز علاوه بر زبان پایهی وب یعنی HTML از زبان SCSS که در واقع CSS با ساختار نوشتاری بهتر و راحت تر است استفاده شده است. همچنین چارچوب انگولار<sup>۴</sup> که تحت توسعه شرکت گوگل است برای این پروژه انتخاب شده است. این چارچوب به دلیل جامعیت و یکپارچگی آن و همچنین استاندارد ساختاری زیبا و یگه و همچنین پشتیبانی شرکت بزرگی چون گوگل گزینه ی مناسبی برای این پروژه به نظر می رسد. در انگولار نیز از زبان تایپاسکریپت به طور پیش فرض استفاده شده است که این امر موجب ارتباط بیشتر و راحت دو بخش بک اند و فرانت اند خواهد شد و تا حدودی از دوباره کاری و بازنویسی جلوگیری می شود.

#### ۱-۴- بررسی نمونه های تجاری مشابه

شاید بتوان گفت نمونه ی مشابهی برای این محصول و با این ساختار وجود نداشته باشد ولی می توان گفت ایده این پروژه به نوعی تلفیق دو محصول خارجی به نام Medium و Reddit است. سرویس اول، بستری برای نوشتن وبلاگ و سرویس دوم بستری برای ساخت انجمن به شیوه ی جدید و با بازطراحی منحصر به فرد است. در این پروژه سعی شده است تا با فراهم کردن بستری یکتا و تعامل گرا در آن، رشد و تکامل وب فارسی بیش از پیش میسر شود.

#### ۱-۵- ساختار پایان نامه

این پایان نامه در ۶ فصل تنظیم شده است که به شرح زیر است:

فصل اول به شرح کلی از هدف پروژه و ساختار پایان نامه می پردازد.

در فصل دوم کلیاتی در مورد چارچوب های مورد استفاده، استاندارد طراحی و زبان های مورد استفاده پرداخته خواهد شد.

---

<sup>۱</sup> MongoDB

<sup>۲</sup> Duplicate

<sup>۳</sup> Front-End

<sup>۴</sup> Angular

فصل سوم به تحلیل و مستندسازی جنبه‌های مختلف پروژه و استفاده از دانش مهندسی نرم‌افزار در طراحی و مدل‌سازی برای توسعه محصول نهایی می‌پردازد.

در فصل چهارم روش‌ها و تکنیک‌های مورد استفاده در توسعه محصول نهایی پروژه، از قبیل چگونگی به‌کارگیری شی‌گرایی و ماژول‌بندی<sup>۱</sup>، نحوه‌ی ایجاد صفحات در یک برنامه‌ی کاربردی و مدیریت حالت‌ها<sup>۲</sup> مورد بررسی قرار خواهد گرفت.

فصل پنجم به نحوه‌ی آزمون نرم‌افزار و بررسی سطوح‌های مختلف آزمون‌های انجام شده از تست یک واحد تا آزمون پذیرش پرداخته خواهد شد و از برآورده شدن نیازمندی‌های اولیه ذکر شده در طول پروژه، اطمینان خواهیم یافت.

در فصل شش به جمع‌بندی فعالیت‌های انجام شده در پروژه و ارائه‌ی دورنمایی از نسخه‌های بعدی محصول می‌پردازد.

---

<sup>1</sup> Modularity

<sup>2</sup> State Management

## فصل دوم

### مفاهیم

#### ۲-۱- مقدمه

در این قسمت به شرح مختصری از برخی مفاهیم کاربردی و تکنولوژی‌ها و زبان‌های انتخاب شده و استاندارد طراحی که در مراحل طراحی و توسعه این نرم‌افزار استفاده شده است خواهیم پرداخت. همچنین در انتها به چگونگی استفاده از متدولوژی انتخاب شده در پیش‌برد این پروژه بیان خواهد شد.

#### ۲-۲- زبان برنامه‌نویسی تایپ‌اسکرپت

این زبان برنامه‌نویسی توسط شرکت مایکروسافت ابتدا در سال ۲۰۱۲ ظاهر شد و آخرین نسخه‌ی آن در حال حاضر نسخه‌ی ۳.۸ است. این زبان، یک زبان متن‌باز با نحو سختگیرانه<sup>۱</sup> است که قابلیت‌های بیشتری نسبت به نسخه‌ی پدر خود یعنی جاوااسکریپت دارد. فلسفه‌ی اصلی مایکروسافت برای ساخت چنین زبانی، وجود نقص‌ها و کاستی‌های جاوااسکریپت در پروژه‌های بزرگ بود که از مهمترین این نقص‌ها می‌توان به «نوع ضعیف»<sup>۲</sup>

---

<sup>۱</sup> Strict Syntactical

<sup>۲</sup> Weakly Typed

بودن و در نهایت هزینه بر بودن نگهداری اشاره نمود. کامپایلر تایپاسکرپت با استفاده از تکنیک بوتاسترپینگ با زبان خودش نوشته شده است که نسخه نهایی به جاوااسکرپت ترجمه می شود. تکنیک بوتاسترپینگ، نوشتن کامپایلر ابتدایی به زبان دیگر و سپس نوشتن کامپایلر نهایی با استفاده از کامپایلر ابتدایی است تا در نهایت کامپایلری با زبان خودمان داشته باشیم. نحو و قابلیت های تایپاسکرپت را می توان ترکیبی از زبان های جاوااسکرپت، جاوا و سی شارپ دانست [۲].

از قابلیت های مهم تایپاسکرپت می توان به موارد زیر اشاره کرد [۲]:

- بررسی نوع در زمان ترجمه
- استنتاج نوع ها
- اینترفیس<sup>۱</sup> ها، نوع های شمارشی<sup>۲</sup> و Generic ها
- قابلیت استفاده از نحو Async/Await
- زنجیربندی اختیاری<sup>۳</sup>

## ۲-۳- چارچوب های نرم افزاری

چارچوب های نرم افزاری از گستره ی وسیعی از کتابخانه ها و ابزارهای کوچک که به طور معمول همه ی آن ها در کنار هم یک برنامه را به وجود می آورند، تشکیل شده است، به طوری که در فرآیند ساخت، تست و بهینه سازی یک محصول می توان از آن ها استفاده نمود. چارچوب ها معمولاً توسط تیم های خبره و با تجربه ساخته می شوند تا علاوه بر قدرتمند بودن و کارآمدی برای توسعه نرم افزارهای کاربردی، تمرکز برنامه نویسی بر روی مورد کاربردهای سطح بالاتر معطوف و از اتلاف وقت در این موارد جلوگیری شود [۳].

## ۲-۳-۱- چارچوب انگولار

این چارچوب که برای توسعه بخش فرانت اند استفاده می شود، اولین نسخه ی آن در سال ۲۰۱۶ توسط شرکت گوگل منتشر شد. این تکنولوژی برای تولید نرم افزار در بستر وب کاربرد دارد. انگولار یک پروژه متن باز است و در نتیجه، علاوه بر تیم انگولار گوگل، توسعه دهندگان دیگر نیز می توانند کدهای آن را مشاهده کنند و بهبود دهند. لازم به ذکر است که این چارچوب نسخه ی از نو نوشته شده AngularJS است که گوگل قبل تر آن را توسعه می داد. همچنین این تکنولوژی را توان از نوع برنامه ی یک صفحه ای (SPA<sup>۴</sup>) نامید.

---

<sup>۱</sup> Interface

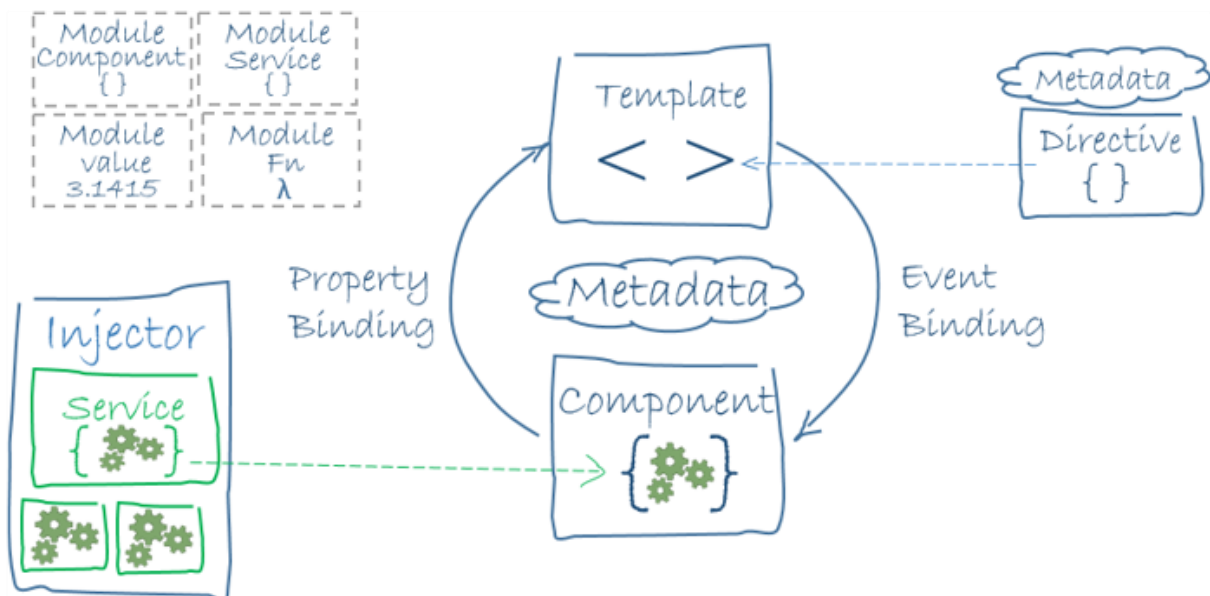
<sup>۲</sup> Enumerated Type

<sup>۳</sup> Optional Chaining

<sup>۴</sup> Single Page Application

از ویژگی‌های مهم انگولار می‌توان به موارد زیر اشاره نمود [۴]:

- استفاده از ساختار سلسله‌مراتبی<sup>۱</sup> مولفه‌ها
- پشتیبانی و توصیه چارچوب به استفاده از زبان تایپ‌اسکرپت
  - نوع‌های ایستا شامل نوع‌های Generic
  - قابلیت استفاده از حاشیه‌نویسی<sup>۳</sup>
  - امکان استفاده از نحو Meta-Programming مانند دکوراتورها
- بارگذاری پویا
- پشتیبانی از اجرا و پردازش انگولار در سرور به جای مشتری (Universal)
- پشتیبانی پیش‌فرض از برنامه‌نویسی رویدادگرا، قدرت گرفته از RxJS



شکل ۱: معماری کلی انگولار [۱]

این چارچوب از نظر قابلیت اطمینان به عدم وجود باگ‌ها و مشکلات، جزو بهترین چارچوب‌های قسمت فرانت‌اند است زیرا قبل از انتشار نسخه‌های جدید، ابتدا گوگل نسخه‌های کاندید انتشار (RC) را بر روی ۱۵۰۰ پروژه خودش تست کرده و در صورت موفقیت آن نسخه بر روی همه‌ی آن‌ها، محصول نهایی را منتشر می‌کند [۵].

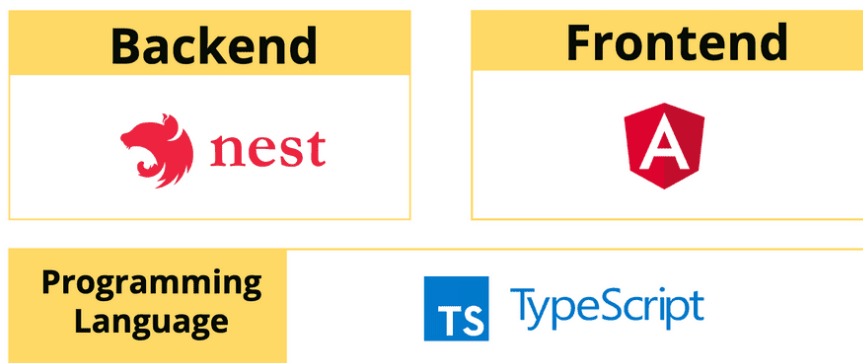
<sup>1</sup> Hierarchy  
<sup>2</sup> Component  
<sup>3</sup> Annotation

## ۲-۳-۲- چارچوب نست جی اس

این چارچوب که بر پایه نودجی اس نوشته شده است هدف اصلی خود را ساخت برنامه های کاربردی کارآمد و مقیاس پذیر بک اند معرفی می کند. این محصول عناصر برنامه نویسی شی گرا، عملکردی<sup>۱</sup> و واکنشی عملکردی<sup>۲</sup> را با هم ترکیب کرده است.

نست جی اس در لایه ی زیرین از چارچوب های قدرتمند سمت سرور مانند اکسپرس (به طور پیش فرض) و فستیفای استفاده می کند. در واقع این محصول سطح انتزاع بالاتری بر روی دیگر چارچوب ها فراهم می آورد. البته این سطح انتزاع بالاتر، از دسترسی به چارچوب های زیرین جلوگیری نمی کند و به توسعه دهنده آزادی استفاده از ماژول های شخص ثالث را نیز می دهد.

فلسفه ی اصلی این چارچوب را می توان در یک کلمه خلاصه نمود و آن «معماری» است. چارچوب های محبوبی چون اکسپرس از معماری خاصی استفاده نمی کنند و این امر در پروژه های بزرگ معضلی جدی خواهد بود و موجب سردرگمی برنامه نویسان و تیم ها خواهد شد و آنان مجبور خواهند بود تا وقت زیادی برای تعیین یک معماری و استاندارد خاص بگذارند. نکته ی جالب توجه برای این چارچوب این است که این تکنولوژی به مقدار زیادی از معماری انگولار الهام گرفته است [۶].



شکل ۲: انگولار و نست جی اس هر دو از زبان تایپ اسکریپت استفاده می کنند.

## ۲-۴- طراحی متریکال برای رابط کاربری / تعاملی<sup>۳</sup>

متریکال دیزاین یک زبان بصری است که اصول کلاسیک یک طراحی خوب را با علوم و تکنولوژی های نوین مانند روانشناسی و علوم شناختی ترکیب کرده است. هدف اصلی آن ایجاد یک سیستم زیربنای واحد برای کاربران است تا ایشان تجربه ی یکسانی از کار با یک برنامه ی کاربردی، بر روی بسترهای مختلف داشته باشند. [۷]

<sup>۱</sup> Functional

<sup>۲</sup> Reactive Functional

<sup>۳</sup> UI/UX



متریال دیزاین برپایه‌ی اصولی که در زیر به آن اشاره کرده‌ایم، بنا شده است [۷]:

(۱) تمامی اجسام در متریال دیزاین، از دنیای واقعی الهام گرفته شده است و در واقع یک استعاره هستند.

(۲) متریال دیزاین بسیار انعطاف‌پذیر طراحی شده است و تمامی عناصر طراحی و افزونه‌های گوناگون در آن به‌طور یکپارچه اجرا می‌شوند.

(۳) با به اشتراک‌گذاری اجزاء و عناصر طراحی خود در میان چارچوب‌ها و بسترهای مختلف، توانسته است واسطه‌ی کاربری یکسانی را میان چارچوب‌های مختلف ایجاد کند. درواقع این زبان یک زبان طراحی چندسکویی است.

## ۲-۵- معماری و ساختار برنامه‌نویسی

همانطور که گفته شد، معماری مورد استفاده در سمت بک‌اند بسیار نزدیک به معماری فرانت‌اند ساخته شده است و با شناخت یکی از آن‌ها، در مدت زمان بسیار کمی می‌توان به شناخت خوبی از تکنولوژی دیگر رسید.

انگولار ساختاری ماژولار دارد. هر ماژول در صورت عدم وجود وابستگی دوار<sup>۱</sup> می‌تواند از ماژول‌ها دیگر استفاده کند. ماژول‌ها عناصر مختلفی را شامل می‌شوند که مهمترین جزء آن مولفه‌ها هستند. هر مولفه به‌طور پیش‌فرض از سه قسمت تشکیل شده‌اند که عبارتند از:

- قالب (template): در واقع همان فایل HTML است.
- طراحی (style): فایل CSS یا هر نوع زبان طراحی دیگر.
- منطق (component): فایل تایپ‌اسکرپت که شامل کلاسی است که ماژول با آن نام مولفه را می‌شناسد.

این ساختار می‌تواند به صورت سلسله‌مراتبی به وسیله استفاده از شناسه‌های هر مولفه در template‌های یکدیگر استفاده شود. همانطور که در شکل ۳ نیز مشخص است در template مولفه‌ی app از مولفه‌هایی چون header و footer استفاده شده است.

```
src > app > app.component.html > ...  
Unsaved changes (cannot determine recent change)  
1 <app-header></app-header>  
2 <router-outlet></router-outlet>  
3 <app-footer></app-footer>  
4
```

شکل ۳: استفاده از مولفه‌ها در دیگر مولفه‌ها

<sup>1</sup> Circular Dependency

نکته‌ی مهمی که در این معماری به طور پیش فرض وجود دارد کپسوله‌بودن قسمت View است. این مفهوم به این معنا است که هر چه در قسمت View مولفه‌ای وجود دارد در دیگر مولفه‌ها قابل دسترسی نخواهد بود تا به این طریق از تغییر ناخواسته مولفه به خاطر مولفه‌های دیگر مصون بماند. به عبارت دیگر این ویژگی باعث حوزه‌بندی مولفه‌ها خواهد شد. برای مثال متغیرهایی که در یک مولفه استفاده می‌شود توسط دیگر مولفه قابل دسترسی نیست [۸].

با توجه به ویژگی قبل مشکلی که به سرعت به ذهن می‌رسد این خواهد بود که بعضی از داده‌ها باید در چندین مولفه استفاده شوند و یا ممکن است درخواست‌های مشابه‌ای از مولفه‌های گوناگون اجرا شود و چگونه می‌توان کدی بهینه‌تر نوشت. راهکار پیش فرضی که معماری انگولار ارائه می‌دهد استفاده از ساختار کلی و تکنیک تزریق وابستگی<sup>۱</sup> است که بخش اصلی آن با نام سرویس‌ها شناخته می‌شوند. به عبارت دقیق‌تر، سرویس‌ها سینگلتون<sup>۲</sup>‌هایی داخل حوزه تزریق شده هستند.

همچنین انگولار از اجزای دیگری مانند محافظ<sup>۳</sup>‌ها، رهگیر<sup>۴</sup>‌ها و لوله<sup>۵</sup>‌ها نیز بهره می‌برد. رهگیرها واسطه‌هایی بین دو موجودیت هستند که معروف‌ترین مورد استفاده‌ی آن واسط بین درخواست‌های HTTP است. محافظ‌ها به بررسی مجوز ورود به صفحات می‌پردازد و لوله‌ها نیز تبدیل‌کننده‌ی داده‌ها به شکل مورد انتظار کاربر در قسمت template گفته می‌شود. همانطور که در شکل ۴ مشخص است در مولفه‌ی HeroBirthday

```
import { Component } from '@angular/core';
@Component({
  selector: 'app-hero-birthday',
  template: `<p>The hero's birthday is {{ birthday | date }}</p>` })
export class HeroBirthdayComponent {
  birthday = new Date(1988, 3, 15); // April 15, 1988
}
// Result: The hero's birthday is Apr 15, 1988
```

شکل ۴: نمونه‌ای از لوله در انگولار

متغیر birthday با مقدار مشخص شده تعریف شده است. در قسمت template و در بخش علامت زده شده از نشان گر « | » استفاده شده و بعد از آن لوله‌ی از قبل تعریف شده‌ی date که کار آن تبدیل شی Date به قالب مناسب برای نمایش به کاربر است استفاده شده است.

بخش مهم دیگر در معماری انگولار مبحث مسیریابی<sup>۶</sup> آن است که می‌توان برای ماژول‌های جدول مسیریابی تعریف کرد و مانند مولفه از ساختار سلسه مراتبی نیز پشتیبانی می‌کند. همچنین می‌توان از ویژگی Lazy Loading نیز در این چارچوب بهره برد به این صورت که ماژول بر اساس نیاز و درخواست کاربر بارگذاری شوند تا رابط کاربری بهتر و سریع‌تری به کاربر ارائه شود.

<sup>۱</sup> Dependency Injection

<sup>۲</sup> Singleton

<sup>۳</sup> Guard

<sup>۴</sup> Interceptor

<sup>۵</sup> Pipe

<sup>۶</sup> Routing

انگولار به هدف سادگی بیشتر از سیستم شناسایی تغییر خودکار بهره می‌برد و به این صورت است که اگر تغییری در مولفه تغییر کند که بر روی DOM اثر بگذارد (حتی کلیک بر روی صفحه) مقدار قبلی متغیر را با مقدار جدید مقایسه کرده و در صورت تغییر آن قسمت را دوباره ترجمه<sup>۱</sup> می‌کند. این روش به دلیل اینکه با هر تغییر در وضعیت مولفه مقدار همه متغیرهای آن دوباره بررسی می‌شود از نظر کارایی جالب به نظر نمی‌رسد. به همین دلیل انگولار راهکار پیشرفته‌تری را ارائه می‌دهد به این صورت که بررسی تغییرات تنها در صورت رخ دادن یک رویداد یا تغییر خواص Input بررسی شده و آن هم فقط آدرس حافظه خانه‌ها بررسی می‌شود. در صورت تغییر مقدارهای داخلی یک متغیر، باید به طور دستی به ترجمه‌گر اطلاع‌رسانی شود.

## ۲-۶- متدولوژی توسعه افزایشی و تکراری

این متدولوژی که زیر مجموعه متدولوژی‌های چابک محسوب می‌شود این گونه تعریف می‌شود:

مدل ساخت افزایشی روندی در مهندسی نرم‌افزار است که مدل به شکل افزایشی طراحی و تست می‌شود (هر دفعه مقدار کمی اضافه می‌شود) تا این که محصول به طور کل کامل تمامی نیازها را برآورده سازد. این متدولوژی ترکیب مدل آبشاری و تکراری است. به بیان دیگر محصول به چندین مولفه شکسته شده و هر مولفه به طور جداگانه ساخته و در صورت تکمیل به محصول کلی اضافه می‌شود [۹].

## ۲-۷- جمع بندی

در این فصل به بررسی مفاهیم موجود و ابزارهای مرتبط به توسعه نرم‌افزار که در این پروژه استفاده می‌شود، پرداخته شد. این مطالب خلاصه‌ای از مفاهیم پایه کارهای انجام شده در طول توسعه این برنامه‌ی کاربردی است. توضیحات اجمالی در صورت نیاز در فصل‌های بعد داده خواهد شد.

---

<sup>1</sup> Render

## فصل سوم

### تحلیل و مستندسازی نرم افزار

#### ۳-۱- مقدمه

در فرایند توسعه‌ی نرم افزار، داشتن درک کاملی از هدف انجام یک پروژه و اطلاع دقیق از نیازمندی‌ها را میتوان اولین و مهمترین گام به‌شمار آورد. در این فصل ابتدا نیازمندی‌های سیستم خود را در قالب سند نیازمندی‌های نرم افزار تبیین نموده و در ادامه نیز با استنتاج موردکاربردها و رسم نمودارهای موردکاربرد، فعالیت و نمودار کلاس طراحی، به تکمیل مستندات خود پرداخته‌ایم.

#### ۳-۲- سند نیازمندی‌های نرم افزار (SRS)

پس از جمع‌آوری نیازمندی‌ها و شناخت قیود و محدودیت‌های موجود، لازم است اطلاعات به‌دست‌آمده به‌طور کامل ثبت شود؛ لذا در صورتی که این اطلاعات در یک قالب مشخص تدوین شود، استفاده آن در آینده بسیار ساده و سریع خواهد بود. به همین جهت در ادامه مطابق با استاندارد SRS 830-1998 IEEE STD که استاندارد برای بیانیه‌ی نیازمندی‌های نرم افزار است؛ به بیان نیازمندی‌های سیستم خود می‌پردازیم [۱۰].

### ۳-۲-۱- هدف

از آنجا که جمع‌آوری نیازمندی‌های کاربران و درک انتظارات آنها از سیستم، گام بسیار مهمی در فرایند تحلیل و طراحی یک پروژه می‌باشد؛ تدوین و نگارش سند نیازمندی‌های نرم‌افزار ضروری خواهد بود.

### ۳-۲-۲- قلمرو

باتوجه به اینکه هدف از انجام این پروژه ایجاد بستری در قالب برنامه تحت وب برای گفتگوهای تخصصی و غیرتخصصی و همچنین بلاگ‌های شخصی با هر موضوعی است می‌توان گفت این پروژه برقراری ارتباط بین تمامی افراد با یکدیگر در سطح کلان خواهد بود ولی از آنجایی که شاید همه‌ی افراد برای گپ و گفت‌وگوهای محاوره‌ای و سرگرمی علاقه‌ای به استفاده از چنین بستری نداشته باشند پس بیشتر قلمروی این محصول به محتوای نظام‌مند و تخصصی معطوف خواهد شد. در نتیجه افراد همان توانایی مشاهده این محتوا را دارند و هم قادر خواهند بود که صفحه‌ی جدید بسازند و آن را مدیریت کنند.

این برنامه‌ی کاربردی با در اختیار گذاشتن ویژگی مختلف برای مدیران صفحات به منظور کاستن از پیچیدگی و هزینه نگهداری و همچنین بهتر شناخته شده آن‌ها بستری مناسب و کارآمد و پویا را برای این قشر به ارمغان می‌آورد. همچنین خوانندگان نیز می‌توانند به عنوان منبع اطلاعاتی و مکان گفتگوی جامع از این بستر استفاده کنند.

### ۳-۲-۳- شرح کلی

این پروژه با هدف بهبود ارتباطات بین کاربران وب فارسی و سهولت مدیریت محتوا و همچنین منبعی قابل دسترسی برای افراد که قابلیت دسترسی از طریق موتورهای جستجوگر را دارد، بنا شده است. با توجه به نیازمندی‌های متغیر افراد و همچنین بر حسب زمان‌های مختلف، افزونه‌هایی به مرور اضافه خواهد شد و در دسترس افراد قرار خواهد گرفت تا آنان بتوانند خود انتخاب کنند که از ابزارهای جدید ارائه شده استفاده کنند یا خیر.

### ۳-۲-۴- کارکرد محصول

مهم‌ترین بخش این برنامه‌ی کاربردی کمک کردن و آسان نمودن ارتباطات رسمی و تخصصی و از طرف دیگر منبعی مناسب برای پرسش‌ها، موضوعات و گفتگوهای کاربران خواهد بود. این بستر با تمرکز بر روی انعطاف‌پذیری و آسانی یادگیری دو نوع از خدمات اطلاع‌رسانی و ارتباطی را فراهم می‌آورد؛ بلاگ و انجمن.

### ۳-۲-۵- مشخصات کاربر

در این پروژه دو نوع کاربری در لایه‌ی بالایی وجود خواهد داشت:

- ۱) صاحبان انجمن‌ها و بلاگ‌ها که در واقع سازنده و مدیر آن صفحه هستند.
- ۲) خوانندگان؛ کاربرانی که مطالب را می‌خوانند و عضو آن انجمن‌ها و بلاگ‌ها می‌شوند.

### ۳-۲-۶- قیود

طراحی این سیستم و توسعه‌ی آن شبیه هر مسئله‌ی بدرفتار دیگری با قیود و محدودیت‌هایی مواجه است. این قیود عبارت‌اند از:

- C1) سیستم باید از سروری قدرتمند جهت بارگذاری سریع اطلاعات و چندرسانه‌ای، استفاده کند.
- C2) سیستم باید قابلیت استفاده در تلفن همراه و کامپیوتر را داشته باشد.
- C3) سیستم باید واکنش‌گرا باشد.
- C4) سیستم باید از اطلاعات شخصی کاربران و حریم خصوصی صفحات محافظت کند.
- C5) سیستم باید محدودیت‌های برای مطالب ارسالی قرار دهد تا موضوعاتی بر خلاف قوانین نظام جمهوری اسلامی درج نشود.

### ۳-۲-۷- مفروضات و نیازمندی‌ها

اتصال دستگاه مورد استفاده (کامپیوتر یا تلفن همراه هوشمند) به اینترنت برای بارگذاری برنامه کاربردی برای نمایش مطالب و گفتگوها ضروری است.

### ۳-۲-۸- نیازمندی‌ها

با بهره‌گیری از روش‌های جمع‌آوری اطلاعات از جمله بررسی پروژه‌ها و محصولات مشابه خارجی، مصاحبه با افراد و اطلاع از نیازهایشان و مطالعه‌ی منابع مفید در این زمینه، به جمع‌آوری نیازمندی‌ها پرداختیم.

### ۳-۲-۸-۱- نیازمندی‌های کارکردی

- R1) سیستم باید امکان ثبت‌نام کاربران را داشته باشد.
- R2) سیستم باید امکان ورود کاربران ثبت‌نام شده را فراهم کند.

- (R3) سیستم باید امکان ساخت انجمن را به کاربران بدهد.
- (R4) سیستم باید امکان ساخت بلاگ را به کاربران بدهد.
- (R5) سیستم باید امکان حذف انجمن را به صاحب آن صفحه بدهد.
- (R6) سیستم باید امکان حذف بلاگ را به صاحب آن صفحه بدهد.
- (R7) سیستم باید امکان ویرایش انجمن را به مدیران صفحه بدهد.
- (R7.1) سیستم باید امکان اضافه کردن، حذف کردن و ویرایش ویجت ها را به مدیران انجمن بدهد.
- (R7.2) سیستم باید امکان ویرایش اطلاعات کلی انجمن را به مدیران بدهد.
- (R7.3) سیستم باید امکان تغییر تم انجمن را به مدیران بدهد.
- (R8) سیستم باید امکان ویرایش بلاگ را به مدیران صفحه بدهد.
- (R8.1) سیستم باید امکان اضافه کردن، حذف کردن و ویرایش ویجت ها را به مدیران بلاگ بدهد.
- (R8.2) سیستم باید امکان ویرایش اطلاعات کلی بلاگ را به مدیران بدهد.
- (R8.3) سیستم باید امکان تغییر تم بلاگ را به مدیران بدهد.
- (R9) سیستم باید امکان ایجاد پست را برای افراد فراهم کند.
- (R10) سیستم باید امکان پاسخ به یک پست را برای افراد فراهم کند.
- (R10.1) سیستم باید امکان پاسخ به شکل یک پست دیگر را محیا کند.
- (R10.2) سیستم باید امکان پاسخ به شکل امتیاز مثبت<sup>۱</sup> و امتیاز منفی<sup>۲</sup> را فراهم کند.
- (R11) سیستم باید امکان اشتراک گذاری یک پست را فراهم کند.
- (R12) سیستم باید امکان تغییر حریم خصوصی که یک بلاگ را بدهد.
- (R13) سیستم باید امکان تغییر حریم خصوصی یک انجمن را بدهد.
- (R14) سیستم باید امکان ویرایش یک پست را به نویسنده آن بدهد.
- (R15) سیستم باید امکان حذف یک پست را به نویسنده و مدیران آن صفحه بدهد.

---

<sup>1</sup> Upvote

<sup>2</sup> Downvote

- R16) سیستم باید امکان گزارش پست با محتوای نامناسب را گزارش را فراهم کند.
- R17) سیستم باید امکان حذف یک پست را به نویسنده و مدیران آن صفحه بدهد.
- R18) سیستم باید امکان مسدود کردن افراد از دیدن یک صفحه توسط مدیر را بدهد.
- R19) سیستم باید امکان آرشیو نمودن پست‌ها را برای کاربران فراهم آورد.
- R20) سیستم باید امکان دسترسی به اطلاعات یک انجمن یا بلاگ را توسط موضوعات و تگ‌های تعریف شده بدهد.
- R21) سیستم باید امکان جستجو بر اساس موضوعات، تگ‌ها و یا اسم را در کل این بستر برای کاربران فراهم کند.
- R22) سیستم باید امکان دنبال کردن صفحات و همچنین موضوعات در کل این بستر را فراهم آورد.
- R23) سیستم باید امکان تغییر گذرواژه در صورت فراموشی آن را به کاربر بدهد.

#### ۳-۲-۸-۲- نیازمندی‌های کارایی

- R1) سیستم باید یک واسط کاربری ساده و زیبا داشته باشد.
- R2) سیستم باید از اطلاعات شخصی کاربران حفاظت کند.
- R3) سیستم باید بتواند به تعداد بسیاری کاربر، سرویس‌دهی کند.
- R4) سیستم باید در بارگذاری صفحات و تصاویر، حداکثر دو ثانیه زمان صرف کند.

#### ۳-۲-۸-۳- قیود طراحی

- C1) هنگام توسعه‌ی برنامه‌ی کاربردی، باید از تصاویر همان اندازه با وضوح بالا و حجم حداکثر یک مگابایت استفاده شود.
- C2) لازم است حجم پکیج نهایی فایل اصلی برنامه به حدی باشد که صفحه در کمتر از پنج ثانیه باز شود.
- C2) لازم است فعالیت‌های مهم، در دید کاربر بوده و زمان دسترسی به آن‌ها بهینه باشد.

#### ۳-۲-۸-۴- صفات سیستم نرم‌افزاری

این سیستم نرم‌افزاری باید شامل صفات زیر باشد:

- **قابلیت استفاده:** برای استفاده از این سیستم کاربران نیاز به آموزش خاصی ندارند؛ زیرا نحوه‌ی طراحی این سیستم به گونه‌ای خواهد بود که افراد در صورت آشنایی با برنامه‌های



کاربردی ارتباطی تحت وب مانند بلاگ‌ها و انجمن‌ها خود بتوانند از این سیستم نیز به‌سادگی استفاده نمایند.

- **قابلیت اطمینان:** به این دلیل که کاربران در هنگام ثبت‌نام، اطلاعات شخصی خود را در سیستم ثبت می‌کنند؛ لازم است به آنها اطمینان داده شود که تمامی اطلاعاتشان محفوظ خواهد ماند.
- **قابلیت دسترسی:** سیستم باید در تمامی زمان‌ها و مکان‌هایی که کاربر به اینترنت متصل است، توانایی سرویس‌دهی سریع به او را داشته باشد.
- در صورتی که کاربر به هر نحو دیگران را مورد اهانت قرار داده و این بستر سوء استفاده کند، سیستم باید به حقوق دیگران احترام گذاشته و به سرعت با چنین افرادی برخورد کند؛ برای مثال حسابشان را مسدود کند.

### ۳-۳- استنتاج مورد کاربردها از نیازمندی‌ها

با استفاده از نیازمندی‌های کارکردی شناخته‌شده در بخش قبل، به استخراج موردکاربردها از نیازمندی‌ها در این مرحله خواهیم پرداخت.

#### ۳-۳-۱- شناسایی مورد کاربردها

در این گام برای شناسایی مورد کاربردها، تمامی نیازمندی‌ها را بررسی نموده و هر عبارت فعلی-اسمی مختص دامنه و همچنین عبارتهای اسمی که کنشگر هستند را پیدا می‌کنیم.

مورد کاربردهای به‌دست‌آمده در این گام در جداول ۳-۱ تا ۳-۲۳ آمده‌اند.

جدول ۱: مورد کاربرد یک (ثبت‌نام کاربر)

مورد کاربرد	اعمال	کنش‌ها
ثبت‌نام کاربر	شروع	از هدر بر روی منو کلیک کنید و بر روی گزینه‌ی «ورود/ثبت‌نام» کلیک کنید.
	هدایت به صفحه‌ی ثبت‌نام	در دیالوگ باز شده بر روی دکمه‌ی ثبت‌نام کلیک کنید.
	وارد کردن اطلاعات	فیلدهای مربوط به ثبت‌نام را به درستی تکمیل کنید.

پس از تکمیل فیلدهای مربوطه، روی دکمه‌ی ثبت‌نام کلیک کنید.	خاتمه	
---	-------	--

جدول ۲: مورد کاربرد دو (ورود کاربر)

کنش‌ها	اعمال	مورد کاربرد
از هدر بر روی منو کلیک کنید و بر روی گزینه‌ی «ورود/ثبت‌نام» کلیک کنید.	شروع	ورود کاربر
فیلدهای مربوط به ورود را به درستی تکمیل کنید.	وارد کردن اطلاعات	
پس از تکمیل فیلدهای مربوطه، روی دکمه‌ی ورود کلیک کنید.	خاتمه	

جدول ۳: مورد کاربرد سه (ساخت انجمن)

کنش‌ها	اعمال	مورد کاربرد
از هدر بر روی منو کلیک کنید و بر روی گزینه‌ی «ساخت انجمن» کلیک کنید.	شروع	ساخت انجمن
فیلدهای مربوط به انجمن را به درستی تکمیل کنید.	وارد کردن اطلاعات	
پس از تکمیل فیلدهای مربوطه، روی دکمه‌ی «ساخت انجمن» کلیک کنید.	خاتمه	

جدول ۴: مورد کاربرد چهار (ساخت بلاگ)

مورد کاربرد	اعمال	کنش‌ها
ساخت بلاگ	شروع	از هدر بر روی منو کلیک کنید و بر روی گزینه‌ی «ساخت بلاگ» کلیک کنید.
	وارد کردن اطلاعات	فیلدهای مربوط به بلاگ را به درستی تکمیل کنید.
	خاتمه	پس از تکمیل فیلدهای مربوطه، روی دکمه‌ی «ساخت بلاگ» کلیک کنید.

جدول ۵: مورد کاربرد پنج (حذف انجمن)

مورد کاربرد	اعمال	کنش‌ها
حذف انجمن	شروع	در صفحه انجمن بر روی چرخ‌دنده کلیک کنید.
	انتخاب دکمه حذف	در تب «اطلاعات کلی» و قسمت پیشرفته بر روی دکمه‌ی «حذف» مربوط به حذف انجمن کلیک کنید.
	خاتمه	در دیالوگ باز شده بر روی «بله» کلیک کنید.

جدول ۶: مورد کاربرد شش (حذف بلاگ)

مورد کاربرد	اعمال	کنش‌ها
حذف بلاگ	شروع	در هدر بلاگ بر روی چرخ‌دنده کلیک کنید.
	ورود به بخش پیشرفته	در منوی سمت راست بر روی «پیشرفته» کلیک کنید.
	انتخاب دکمه‌ی حذف	بر روی دکمه «حذف» در ردیف حذف بلاگ کلیک کنید.

خاتمه	در دیالوگ باز شده بر روی «بله» کلیک کنید.
-------	---

جدول ۷: مورد کاربرد هفت (شروع ویرایش انجمن)

مورد کاربرد	اعمال	کنش‌ها
شروع ویرایش انجمن	شروع	در هدر انجمن بر روی چرخ‌دنده کلیک کنید.
	انتخاب قسمت تغییرات	بر روی تب مورد نظر که قصد تغییر را دارید کلیک کنید.

جدول ۸: مورد کاربرد هشت (ویرایش اطلاعات کلی انجمن)

مورد کاربرد	اعمال	کنش‌ها
ویرایش اطلاعات کلی انجمن	شروع	فیلدها و دکمه‌های موجود در صفحه را مشاهده کنید.
	وارد کردن اطلاعات	فیلدهای مورد نظر را با مقدار جدید جایگزین کنید.
	خاتمه	بر روی دکمه‌ی «ذخیره تغییرات» کلیک کنید.


جدول ۹: مورد کاربرد نه (ویرایش یک ویجت)

مورد کاربرد	اعمال	کنش‌ها
ویرایش یک ویجت	شروع	در تب ویجت‌ها ویجت‌های فعال را مشاهده کنید.
	انتخاب دکمه ویرایش	در جعبه ویجت مورد نظر بر روی دکمه با شکل  کلیک کنید.
	وارد کردن اطلاعات	فیلدهای مورد نظر را با مقدار جدید جایگزین کنید.
	خاتمه	بر روی دکمه‌ی «ذخیره تغییرات» کلیک کنید.

جدول ۱۰: مورد کاربرد ۱۰ (ویرایش ترتیب ویجت‌ها)

مورد کاربرد	اعمال	کنش‌ها
ویرایش ترتیب ویجت‌ها	شروع	در تب ویجت‌ها ویجتی که می‌خواهید جایگاهش عوض شود را با کلیک چپ بگیرید.
	وارد کردن اطلاعات	در همان حالت، ویجت را به جایگاهی که می‌خواهید بکشید و رها کنید.
	خاتمه	بر روی دکمه‌ی «ذخیره تغییرات» کلیک کنید.

جدول ۱۱: مورد کاربرد ۱۱ (حذف ویجت)

مورد کاربرد	اعمال	کنش‌ها
حذف ویجت	شروع	در تب ویجت‌ها، ویجت‌های فعال را مشاهده کنید.
	کلیک بر روی دکمه‌ی حذف	در جعبه ویجت مورد نظر بر روی دکمه با شکل  کلیک کنید.
	خاتمه	بر روی دکمه‌ی «ذخیره تغییرات» کلیک کنید.

جدول ۱۲: مورد کاربرد ۱۲ (ویرایش تم)

مورد کاربرد	اعمال	کنش‌ها
ویرایش تم	شروع	در تب اطلاعات کلی تنظیمات، به قسمت رنگ‌ها مراجعه کنید
	انتخاب قسمت رنگ‌بندی	بر روی دایره رنگی مربوط کلیک کنید.
	انتخاب رنگ	رنگ مورد نظر خود را انتخاب کنید.
	خاتمه	بر روی «ذخیره تغییرات» کلیک کنید.

جدول ۱۳: مورد کاربرد ۱۳ (ایجاد پست جدید)

کنش‌ها	اعمال	مورد کاربرد
در انجمن یا بلاگ مربوط بر روی «پست جدید» کلیک کنید.	شروع	ایجاد پست
موضوع پست را وارد کنید.	انتخاب عنوان پست	
به طور اختیاری می‌توانید زیرنویس موضوع را بنویسید.	انتخاب زیرنویس	
در قسمت مشخص شده می‌توانید متن پست را بنویسید و در صورت نیاز با کلیک بر روی سه نقطه از امکانات ویرایش استفاده کنید.	نوشتن متن پست	
در لیست زیرموضوعات می‌توانید زیرموضوعات مربوط به پست را وارد کنید.	انتخاب زیرموضوعات	
بر روی دکمه انتشار کلیک کنید.	خاتمه	

جدول ۱۴: مورد کاربرد ۱۴ (پاسخ متنی به یک پست)

کنش‌ها	اعمال	مورد کاربرد
در قسمت پست مورد نظر بر روی «نظرات» کلیک کنید.	شروع	پاسخ متنی به یک پست
در صفحه‌ی پست در بخش نظرات و فیلد مربوطه متن خود را وارد کنید.	انتخاب قسمت رنگ‌بندی	
بر روی «ثبت نظر» کلیک کنید.	خاتمه	

جدول ۱۵: مورد کاربرد ۱۵ (امتیاز دادن به یک پست)

کنش‌ها	اعمال	مورد کاربرد
در قسمت پست مورد نظر دکمه‌ی بالا (👍) و پایین (👎) را بیاورید.	شروع	امتیاز مثبت یا منفی دادن به یک پست

بر روی گزینه مورد نظرتان کلیک کنید.	خاتمه	
-------------------------------------	-------	--

جدول ۱۶: مورد کاربرد ۱۶ (ویرایش یک پست)

کنش‌ها	اعمال	مورد کاربرد
در قسمت پست مورد نظر دکمه‌ی سه نقطه در سمت چپ بالا را بزنید و روی ویرایش کلیک کنید.	شروع	ویرایش یک پست
در صفحه‌ی باز شده مقادارها را به روزرسانی کنید.	تغییر محتوای پست	
بر روی دکمه‌ی «بروزرسانی» کلیک کنید.	خاتمه	

جدول ۱۷: مورد کاربرد ۱۷ (حذف یک پست)

کنش‌ها	اعمال	مورد کاربرد
در قسمت پست مورد نظر دکمه‌ی سه نقطه در سمت چپ بالا را بزنید و روی حذف کلیک کنید.	شروع	حذف یک پست
در دیالوگ باز شده بر روی «بله» کلیک کنید	خاتمه	

جدول ۱۸: مورد کاربرد ۱۸ (تغییر مشخصات کاربری)

کنش‌ها	اعمال	مورد کاربرد
در قسمت هدر بر روی منو کلیک کنید و در منوی باز شده بر روی «تنظیمات کاربری» کلیک کنید.	شروع	تغییر مشخصات کاربری

انتخاب تب مورد نظر	در صفحه‌ی باز شده به تب «پروفایل» یا «حساب کاربری» بر حسب نیاز بروید.
انتخاب مقدار فیلدها	مقدار جدید را بر حسب فیلدهای مورد نظر وارد کنید.
خاتمه	بر روی «ذخیره تغییرات» کلیک کنید.

جدول ۱۹: مورد کاربرد ۱۹ (تغییر حریم خصوصی کاربر)

مورد کاربرد	اعمال	کنش‌ها
تغییر حریم خصوصی کاربر	شروع	در قسمت هدر بر روی منو کلیک کنید و در منوی باز شده بر روی «تنظیمات کاربری» کلیک کنید.
	انتخاب تب حریم خصوصی	در صفحه‌ی باز شده به تب «حریم خصوصی» بروید.
	انتخاب مقدار فیلدها	مقدار دلخواه را بر اساس ردیف مدنظر تغییر دهید.
	خاتمه	بر روی «ذخیره تغییرات» کلیک کنید.

جدول ۲۰: مورد کاربرد ۲۰ (گزارش محتوای نامناسب)

مورد کاربرد	اعمال	کنش‌ها
گزارش محتوای نامناسب	شروع	در قسمتی که به نظرتان محتوای نامناسبی دارد (پست، بلاگ یا انجمن) بر روی سه نقطه کلیک کنید و دکمه «گزارش» را بزنید.
	انتخاب علت نامناسب بودن	در دیالوگ باز شده موارد نامناسب را انتخاب کنید و توضیح خود را در صورت نیاز بنویسید.



بر روی «ارسال گزارش» کلیک کنید.	خاتمه	
---------------------------------	-------	--

جدول ۲۱: مورد کاربرد ۲۱ (جستجو در تمام سایت)

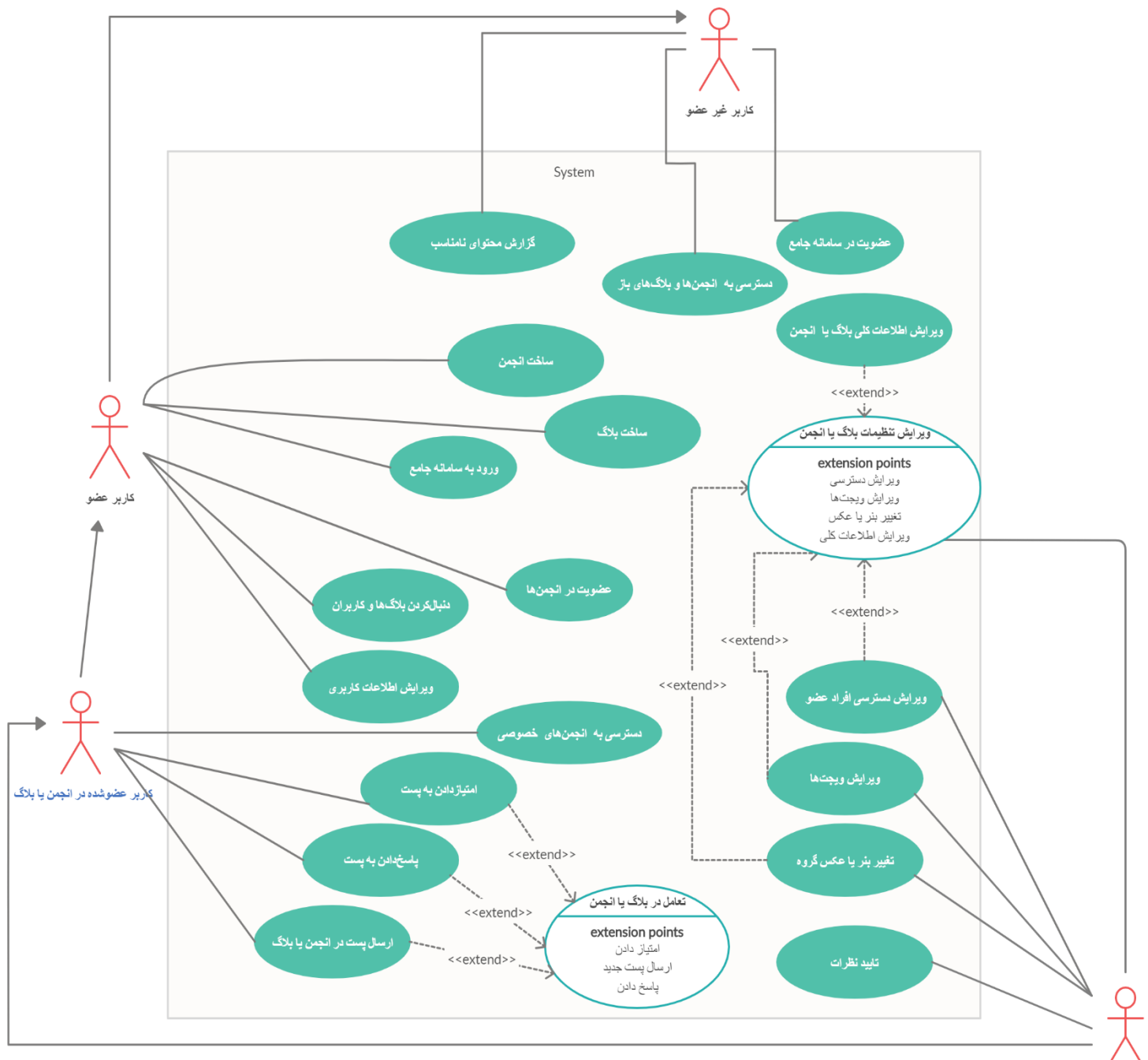
کنش‌ها	اعمال	مورد کاربرد
بر روی دکمه ذره‌بین کلیک کنید.	شروع	جستجو در تمام سایت
در فرم باز شده فیلدهای مورد جستجویتان را وارد کنید.	انتخاب علت نامناسب بودن	
بر روی نتایج لحظه‌ای کلیک کنید.	خاتمه	

جدول ۲۲: مورد کاربرد ۲۲ (عضویت یا دنبال کردن)

کنش‌ها	اعمال	مورد کاربرد
وارد صفحه‌ی مورد نظر شده و در اولین جعبه سمت چپ را مشاهده کنید.	شروع	عضویت در انجمن‌ها یا دنبال کردن بلاگ‌ها
بر روی دکمه «دنبال کردن» یا «عضویت» کلیک کنید.	خاتمه	

### ۳-۲-۳- مصور سازی مورد کاربردها

درموردی که تعداد موردکاربردها زیاد است، نیاز به مصورسازی و سازماندهی این اطلاعات وجود دارد. به همین دلیل از نمودار موردکاربردها که یک نمودار رفتاری یوامال<sup>۱</sup> است برای نمایش موردکاربردهای سیستم فعلی استفاده کرده و نتیجه‌ی حاصل از آن در شکل ۵ به نمایش درآمده است.



شکل ۵: نمودار مورد کاربردها

<sup>۱</sup> UML (Unified Modeling Language)

### ۳-۴- مدل سازی تعامل کنش گر - سیستم

مدل سازی تعامل کنشگر - سیستم عبارت است از مدل سازی و طراحی این که چگونه سیستم با کنش گر ها تعامل می کند تا یک مورد کاربرد انجام شود [۱۰].

در اولین گام از مدل سازی تعامل کنشگر-سیستم، لازم است برای هریک از موردکاربردها جداول دوستونی تشکیل داده و نحوه ی تعامل آنها با سیستم را شرح دهیم [۱۰]. در ادامه جداول ۲۳ که به عنوان نمونه برای یک مورد از موردکاربردهای ذکرشده در قسمت قبل تنظیم شده است، مشاهده می شود.

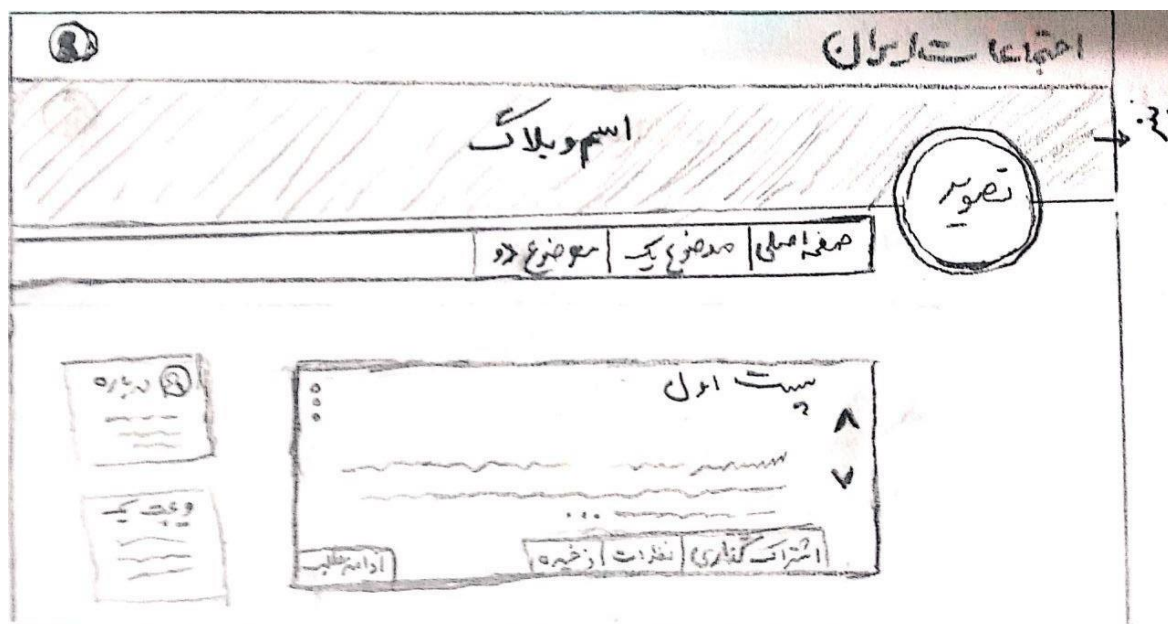
جدول ۲۳: جدول دو ستونی مورد کاربرد ۱۸ (تغییر مشخصات کاربری)

کنشگر: کاربر	سیستم: وب اپلیکیشن
	۰ - برنامه صفحه ی تغییر مشخصات را به کاربر نمایش می دهد.
۱ - TUCBW کاربر تب های مختلف را بالای صفحه مشاهده می کند و گزینه ی مورد نظرش را انتخاب می کند.	۲ - برنامه به رندر کردن صفحه مربوط مشغول می شود و اطلاعات مربوط در صورت نیاز را از سمت سرور می گیرد و به کاربر نمایش می دهد.
۳ - کاربر با مشاهده ی صفحه، فیلد یا فعالیت مورد نظر را فعال می کند.	۴ - برنامه با توجه به عملی که کاربر انجام داد بازخورد مناسبی را به کاربر نمایش می دهد تا کاربر از صحت انجام کار خود مطمئن شود.
۴ - کاربر مقدار جدید را قبلی جایگزین می کند.	۵ - برنامه مقدار جدید را در همان قسمت به کاربر نمایش می دهد.
۶ - کاربر پس انجام تغییرات لازم بر روی دکمه «ذخیره تغییرات» کلیک می کند.	۷ - برنامه به بررسی معتبر بودن اطلاعات وارد شده توسط کاربر می پردازد. در صورت صحیح بودن اطلاعات، آن را به سمت سرور ارسال می کند و نتیجه ذخیره اطلاعات را به کاربر نمایش می دهد.
۸ - TUCEW کاربر پیغام متناسب با نتیجه ارسال را مشاهده میکند.	

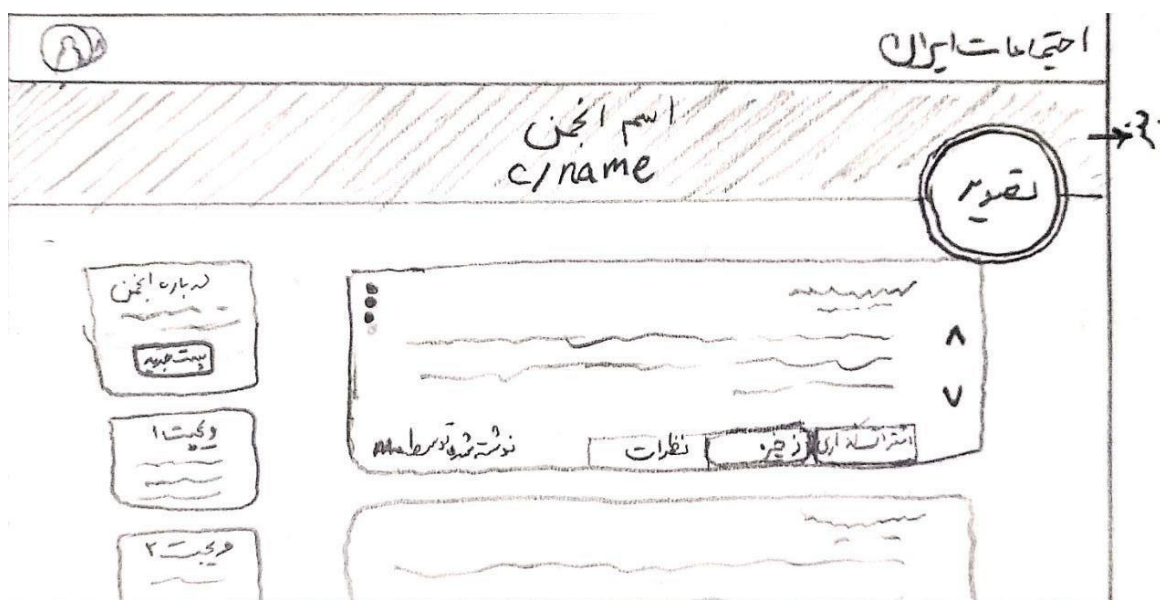
برای سایر موردکاربردها نیز به همین ترتیب جداول دوستونی تنظیم می شود تا نحوه ی تعامل کاربر با سیستم برای رفع تمامی نیازمندی های او به طور دقیق و کامل قابل تشخیص باشد.

### ۳-۴-۱- استفاده از پیش‌نمونه‌های واسط کاربری

در این بخش به‌منظور تفهیم هرچه بیشتر سیستم موردنظر، تعدادی از واسط‌های کاربری موردانتظار، به‌تصویر کشیده‌شده‌اند. این واسط‌ها به ترتیب در شکل‌های ۷ تا ۱۲ قابل مشاهده هستند.



شکل ۷: صفحه اصلی بلاگ



شکل ۶: صفحه اصلی انجمن

احتمالات ایران

اطلاعات کلی | ویژگیها | مخدوم رستری

عمری

عنوان انجمن

به تاروی انجمن

زیر مخدومات

تصویر

انتخاب

تصویر

شکل ۸: تنظیمات انجمن - اطلاعات کلی (قسمت یک)

احتمالات ایران

تصویر

انتخاب

رنگ

رنگ پس زمینه

رنگ متن

رنگ

رنگ

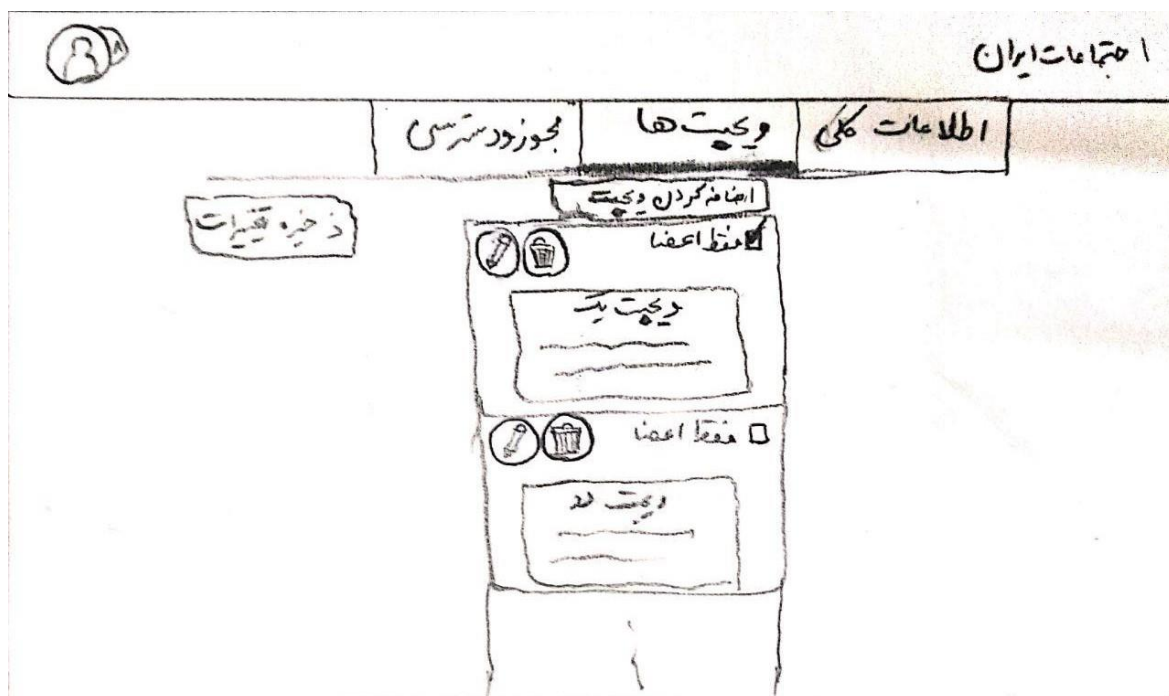
پس زمینه

انجمن خصوصی

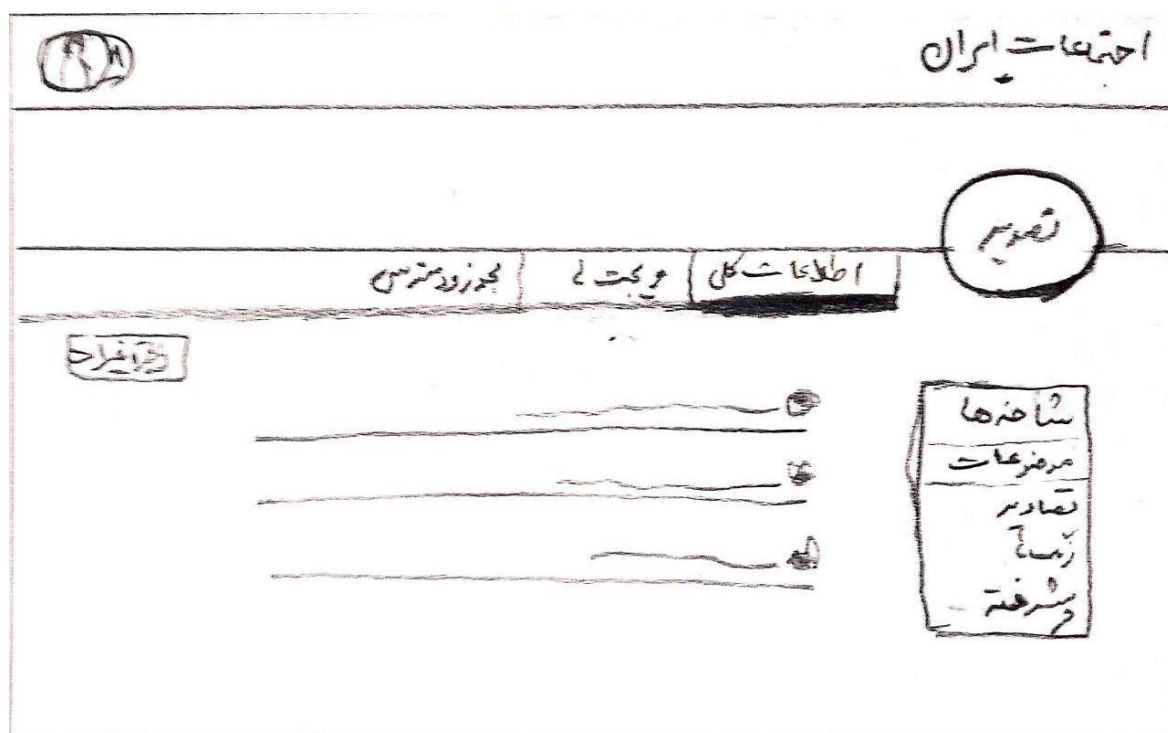
حذف انجمن

حذف

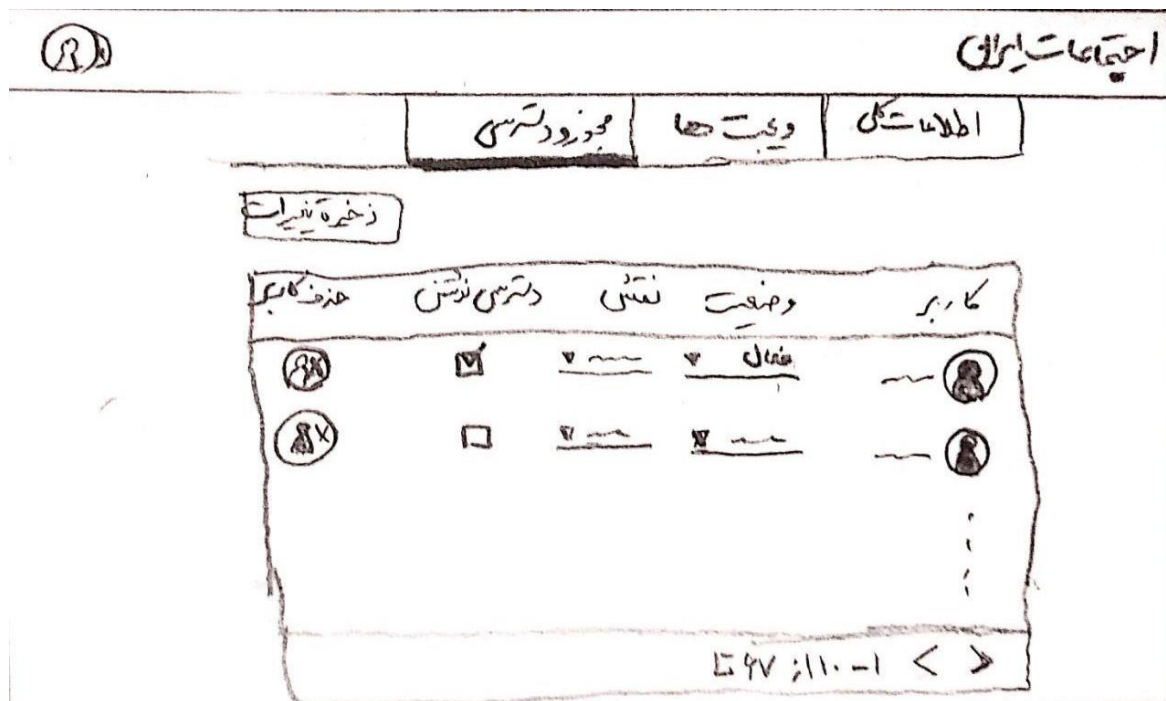
شکل ۹: تنظیمات انجمن - اطلاعات کلی (قسمت دو)



شکل ۱۱: تنظیمات ویجت ها



شکل ۱۰: تنظیمات وبلاگ - اطلاعات کلی



شکل ۱۲: تنظیمات انجمن - مجوزها و دسترسی

### ۳-۵- مدلسازی تعامل شیء

در این قسمت به مدلسازی تعامل شیء می‌پردازیم که عبارت است از فرایندی که [۱۰]:

- به درک این موضوع که در سیستم جاری یا در فرایندهای کسب‌وکار جاری چگونه اشیا باهم تعامل میکنند، کمک می‌کند.
- سبب شناسایی مشکلات و محدودیت‌های سیستم یا فرایندهای کسب و کار جاری می‌شود.
- تعیین خواهد کرد که در سیستم نرم‌افزاری پیشنهادی چگونه اشیا با هم تعامل داشته باشند تا موردکاربردها را به‌انجام برسانند.

### ۳-۵-۱- شناسایی گام‌های غیر بدیهی

موردکاربردهای گسترده که در بخش قبل معرفی شدند، جداول دوستونی بودند که چگونگی تعامل یک کنشگر با سیستم را نشان می‌دهند. در یکی از گام‌های ذکرشده در این جداول دوستونی، کنشگر از سیستم می‌خواهد که پردازشهای پس‌زمینه‌ای انجام دهد. به این معنی که، اشیاء نرم‌افزاری را وادار کند باهم تعامل و همکاری داشته باشند و درخواست موردنظر را برآورده نمایند. به چنین درخواستی، یک درخواست غیربدیهی گفته می‌شود [۱۰].



حال به عنوان نمونه گام غیربديهی شناسایی شده برای مورد کاربرد ۱۸، که در جدول ۲۳ در بخش قبل آمده است، با علامت \*\* در جدول ۲۴ مشاهده می شود.

جدول ۲۴ گام غیربديهی شناخته شده برای مورد کاربرد ۱۸ (تنظیمات مشخصات کاربری)

کنشکر: کاربر	سیستم: وب اپلیکیشن
	۰ - برنامه صفحه ی تغییر مشخصات را به کاربر نمایش می دهد.
۱ - TUCBW کاربر تب های مختلف را بالای صفحه مشاهده می کند و گزینه ی مورد نظرش را انتخاب می کند. **	۲ - برنامه به رندر کردن صفحه مربوط مشغول می شود و اطلاعات مربوط در صورت نیاز را از سمت سرور می گیرد و به کاربر نمایش می دهد.
۳ - کاربر با مشاهده ی صفحه، فیلد یا فعالیت مورد نظر را فعال می کند.	۴ - برنامه با توجه به عملی که کاربر انجام داد بازخورد مناسبی را به کاربر نمایش می دهد تا کاربر از صحت انجام کار خود مطمئن شود.
۴ - کاربر مقدار جدید را با قبلی جایگزین می کند.	۵ - برنامه مقدار جدید را در همان قسمت به کاربر نمایش می دهد.
۶ - کاربر پس انجام تغییرات لازم بر روی دکمه ذخیره تغییرات کلیک می کند. **	۷ - برنامه به بررسی معتبر بودن اطلاعات وارد شده توسط کاربر می پردازد. در صورت صحیح بودن اطلاعات، آن را به سمت سرور ارسال می کند و نتیجه ذخیره اطلاعات را به کاربر نمایش می دهد.
۸ - TUCEW کاربر پیغام متناسب با نتیجه ارسال را مشاهده می کند.	



### ۳-۵-۲- نوشتن سناریو برای گام غیربديهی

پس از شناسایی گام‌های غیربديهی، لازم است سناریوهای مربوط به این گام‌ها تبیین شود. سناریوها در واقع دنباله‌ای از جملات اعلانی هستند که می‌گویند اشیا چگونه با یکدیگر تعامل می‌کنند تا یک گام غیربديهی انجام شود [۸].

سناریوی مربوط به گام غیربديهی شناسایی شده در بخش ۳-۵-۱ در شکل ۱۳ آمده است.

۶) کاربر پس انجام تغییرات لازم بر روی دکمه «ذخیره تغییرات» کلیک می‌کند.

۷.۱) در مولفه مربوط به آن قسمت تابع مربوطه به بخش صدا زده می‌شود.

۷.۲) در تابع به اعتبارسنجی مقدار وارد شده پرداخته می‌شود و در صورت معتبر نبودن پیغام مناسب نشان داده می‌شود. در صورت صحیح بودن به قسمت ۷.۳ می‌رویم.

۷.۳) داده‌های مورد نیاز به بخش مدیریت وضعیت UserAction مربوطه فرستاده می‌شود.

۷.۴) مقداردهی‌های لازم در قسمت اطلاعات کاربری سمت فرانت‌اند انجام شود. (loading مقدار true می‌گیرد)

۷.۵) در قسمت Effect بخش مدیریت وضعیت کاربر بسته ارسالی آماده شده و به سمت سرور فرستاده می‌شود.

۷.۶) پاسخ ارسالی سرور بررسی می‌شود و در صورت موفقیت‌آمیز بودن، مقدار جدید در فرانت‌اند نیز قرار می‌گیرد و پیام مربوطه به کاربر نمایش داده شده و loading مقدار false می‌شود. در صورت خطا بخش ۷.۷ اجرا می‌شود.

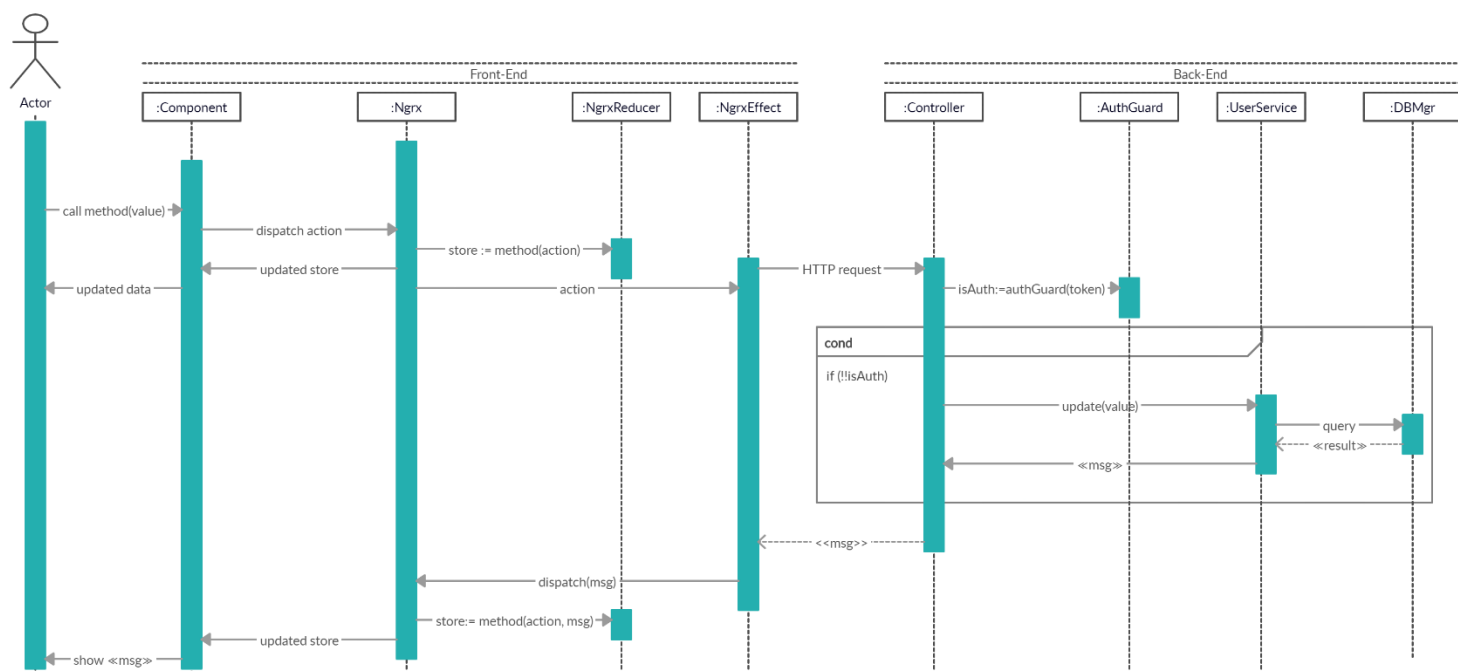
۷.۷) خطای به وجود آمده به کاربر نمایش داده می‌شود و loading مقدار false می‌گیرد.

شکل ۱۳ سناریو گام غیر بديهی مورد کاربر ۱۸ (تغییر مشخصات کاربری)

### ۳-۵-۳- استنتاج نمودارهای توالی از سناریو

در این گام سناریوهای نوشته شده در بخش قبل به نمودارهای توالی تبدیل خواهند شد تا از خوانایی بیشتری برخوردار بوده و استفاده از آن‌ها در آینده ساده‌تر شود.

برای نمونه، سناریوی نوشته شده برای مورد کاربرد ۱۸ در بخش قبل به نمودار توالی تبدیل شده و در شکل ۱۴ آمده است.



شکل ۱۴: نمودار توالی مورد کاربرد ۱۸ (تغییر مشخصات کاربری)

### ۳-۶- استنتاج نمودار کلاس طراحی

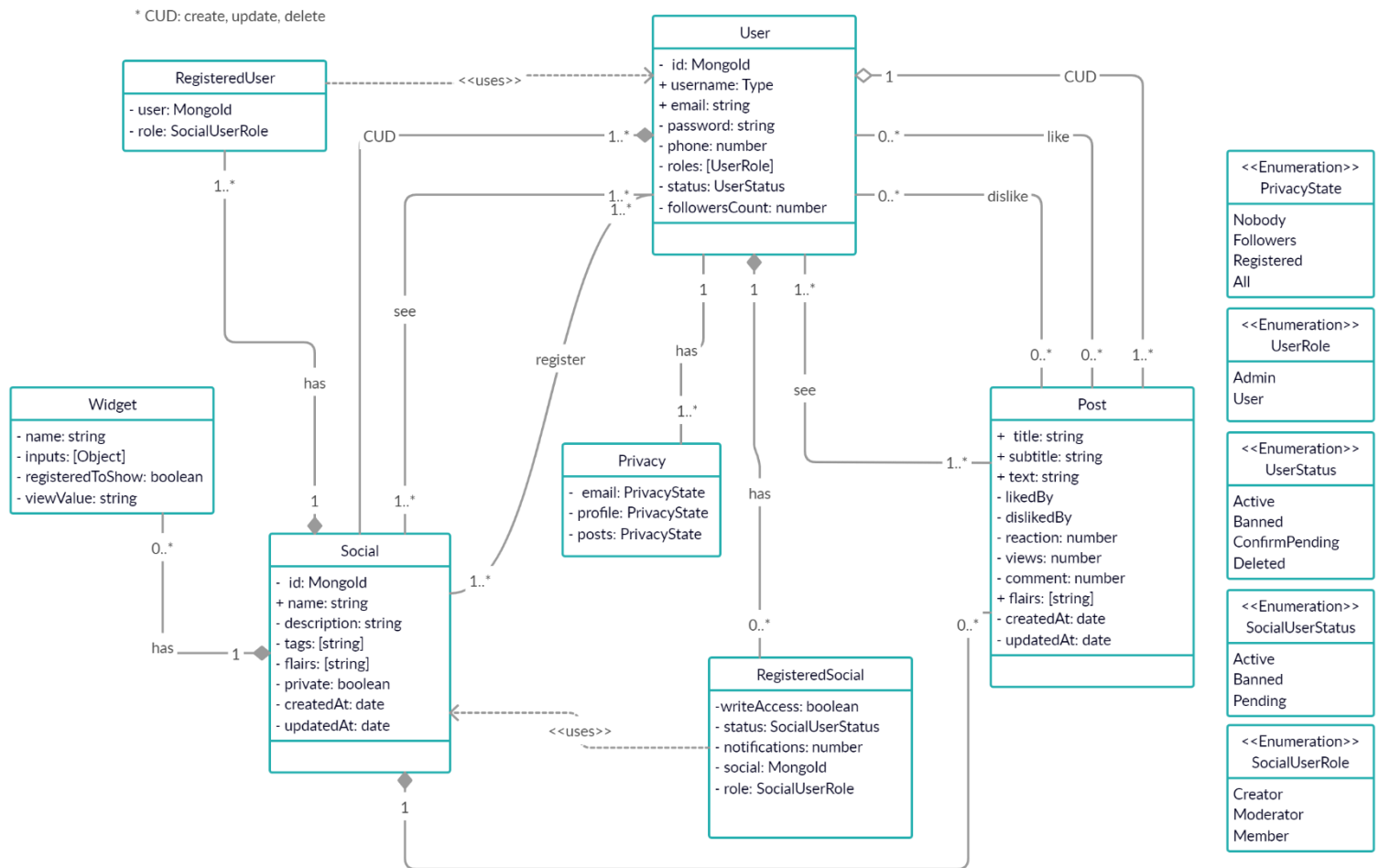
پروژه‌های دنیای واقعی نیازمند کلاس‌ها، توابع و ویژگی‌های آنها و روابط بین کلاس‌ها هستند و اگر کلاس‌ها یا اعمال نادیده گرفته شوند، درحین آزمون یکپارچه‌سازی مشکلات، خود را نشان خواهند داد.

نمودارهای UML تولیدشده در گام‌های قبلی، جنبه‌های مختلف کاربرد سیستم در دست ساخت را توصیف می‌کنند. ولی مجموعه‌ای این نمودارها یک ساختار کلاس و یک نقشه‌ی راه که بتواند راهنمایی برای توسعه‌ی آزمون-راننده و آزمون یکپارچه‌سازی باشد را فراهم نمی‌سازد. بنابراین ما نیازمند نموداری هستیم که به‌عنوان مشخصات طراحی کلاس‌ها و ساختار کلاسی عمل کند.

نمودار کلاس DCD یک نمودار کلاس UML است که از روی مدل‌های رفتاری شامل نمودارهای توالی، حالت و فعالیت و مدل دامنه به‌دست‌آمده است. نمودار کلاس طراحی یک نقشه‌ی طراحی است که فعالیت‌های بعدی پیاده‌سازی، آزمون و یکپارچه‌سازی را تسهیل می‌کند [۸].

### ۳-۶-۱ - به تصویر کشیدن نمودار کلاس طراحی

پس از شناسایی کلاس‌ها، متدها، صفات‌ها و روابط که پیش‌زمینه‌ی رسم نمودار کلاس طراحی هستند، نتیجه‌ی حاصل از رسم این نمودار در شکل ۱۵ مشاهده می‌شود.



شکل ۱۵: نمودار کلاس

### ۳-۷ - جمع‌بندی

در این فصل به گردآوری مطالبی پرداختیم که فهم و نگهداری محصول را ساده‌تر سازد و در آینده به‌عنوان مرجعی معتبر برای پروژه‌های مشابه مورد استفاده قرار گیرد.

## فصل چهارم

### پیاده‌سازی

#### ۴-۱ - مقدمه

در این بخش به معرفی برخی از مهمترین جنبه‌های تکنیکی و نحوه‌ی پیاده‌سازی پروژه خواهیم پرداخت. آشنایی با چگونگی استفاده از چارچوب انگولار و نست‌جی‌اس، ایجاد قسمت‌های مختلف مانند مولفه، سرویس، استور، کنترلر، گارد و نمایش خروجی اصلی آن‌ها، از مهمترین سرفصل‌هایی هستند که در ادامه به تفصیل به بیان آن‌ها پرداخته شده است.

#### ۴-۲ - چارچوب انگولار

در این قسمت نحوه‌ی ساخت یک برنامه‌ی وب ساده در انگولار، ساختار پروژه، مکانیزم‌های اتصال داده، انواع بارگذاری باندل‌ها بر حسب نیاز، مدیریت وضعیت و محافظ‌ها در این چارچوب پرداخته خواهد شد.

#### ۴-۲-۱ - استفاده از چارچوب انگولار

برای استفاده از چارچوب انگولار و توسعه‌ی برنامه‌های کاربردی به کمک آن، روش‌های گوناگونی وجود دارد. اما اصلی‌ترین روش آن، استفاده از مخزن npm است. پس از بارگذاری و

```
> yarn global add @angular-cli
```

نصب نرم افزار yarn بر روی کامپیوتر در خط فرمان دستور زیر برای نصب انگولار انجام می شود:

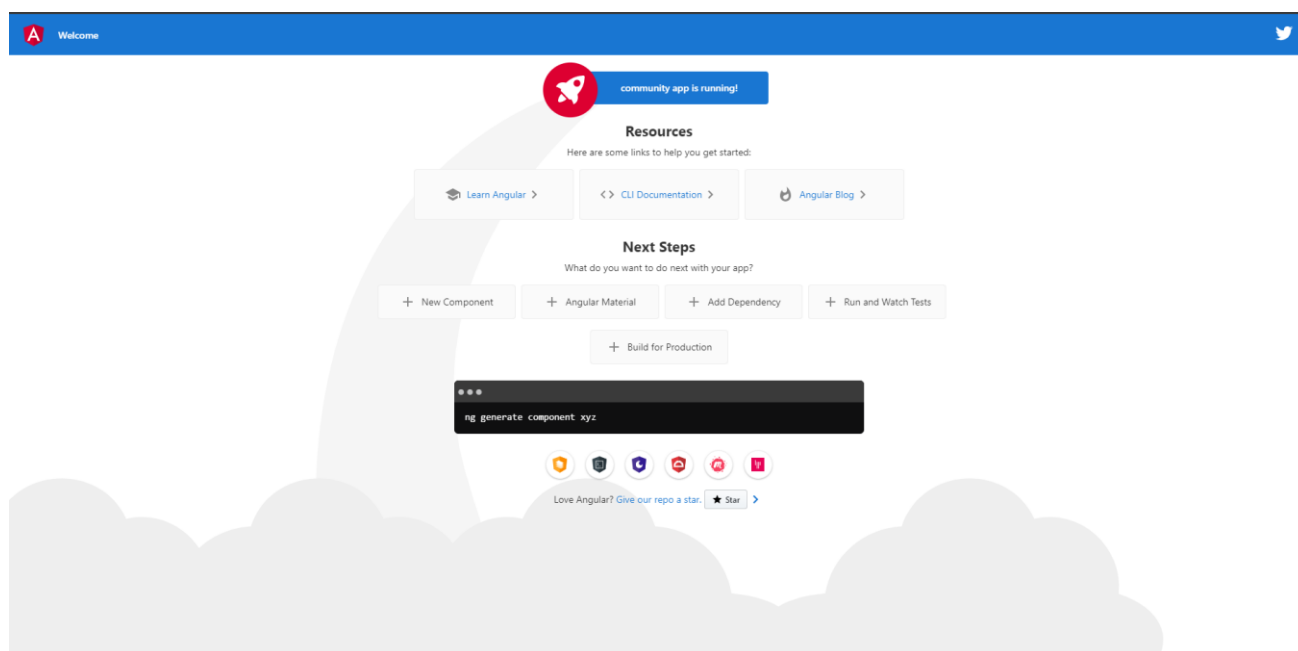
بعد از به اتمام رسیدن نصب انگولار، به ساخت یک پروژه ی جدید می پردازیم:

```
> ng new community --style=scss --routing=true
```

با دستور بالا پروژه ی جدیدی با نام «community»، پیش پردازش گر استایل «scss» و ایجاد وابستگی های فایل مسیریابی ایجاد می شود. برای تست اجرای وب اپلیکیشن از دستور زیر استفاده می شود که برنامه ی کاربردی را به طور پیش فرض در لوکال هاست با پورت ۴۲۰۰ ایجاد خواهد کرد.

```
> ng serve -o
```

که نتیجه به شکل زیر خواهد بود:



شکل ۱۶: اولین پروژه ی انگولار

برای اضافه کردن کتابخانه های لازم به پروژه دو روش کلی وجود دارد. روش اول که بیشتر پیشنهاد می شود استفاده از دستور زیر است:

```
> yarn add @angular/material
```

این دستور با استفاده از شماها انگولار و اسکریپت های موجود علاوه بر نصب پکیج، تغییرات لازم را در پروژه به وجود می آورد تا نیاز نباشد به طور دستی، پروژه را برای کار با این کتابخانه آماده کنیم.

روش دوم به صورت زیر است:

```
> ng add @angular/material
```

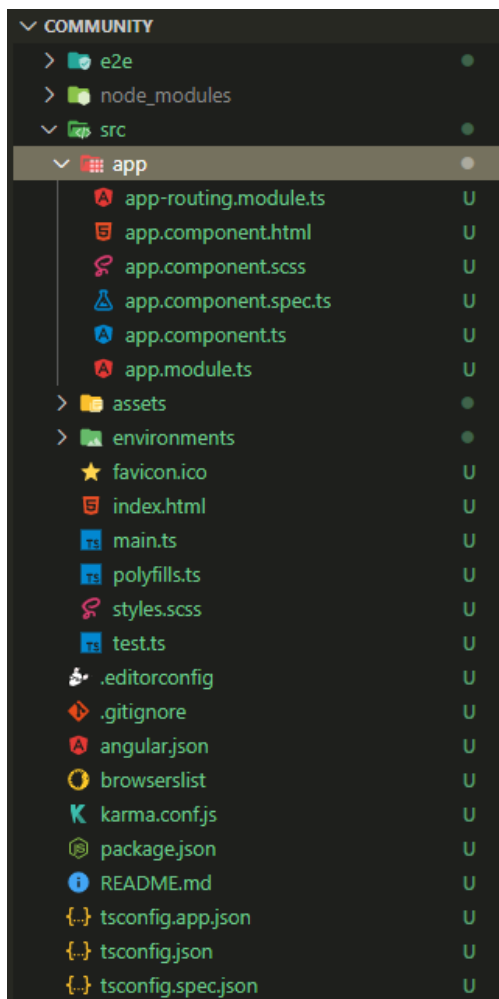
در این روش تنها پکیج مورد نظر نصب می‌شود و تغییرات مورد نیاز باید دستی انجام شود. لیست پکیج‌های موردنیاز برای این پروژه در پیوست ۱ قرار گرفته است.

محیط توسعه انتخاب شده برای برنامه‌نویسی نیز می‌تواند ویژوال استودیو<sup>۱</sup>، وب‌استورم<sup>۲</sup>، وی‌اس‌کد<sup>۳</sup> و یا دیگر محیط‌های توسعه استفاده نمود که در این پروژه از وی‌اس‌کد برای توسعه کل پروژه استفاده شده است. افزونه‌های استفاده شده برای محیط توسعه در پیوست ۲ قرار گرفته است.

## ۴-۲-۲- ساختار پروژه‌ها در انگولار

هنگامی که با دستوری که بالا گفته شده پروژه‌ی جدید انگولار ساخته می‌شود،

فایل‌هایی به وجود می‌آیند و یه ساختار کلی شکل می‌گیرد. این ساختار در شکل ۱۷ دیده می‌شود.



همانطور که مشاهده می‌شود، در قسمت ریشه پروژه سه فولدر وجود دارد. e2e که فایل‌های تست end-to-end در آن قرار می‌گیرد. فولدر node\_modules مکان نصب کتابخانه‌های پروژه است. فولدر src هم مکان قرارگیری کدهای وب‌اپلیکیشن و فعالیت‌های اصلی در آن قرار می‌گیرد. فایل tslint قواعد نگارشی تایپ‌اسکرپت توصیه‌شده توسط تیم انگولار است که با نصب افزونه‌ی مربوط به آن بر روی محیط توسعه می‌توان از راهنمای نوشتار آن استفاده نمود. tsconfig هم پیکربندی تایپ‌اسکرپت است. فایل angular.json نیز پیکربندی محیط توسعه و نهایی پروژه‌ی انگولار است.

در فولدر src فایل‌های index.html و main.ts اولین فایل‌هایی هستند که در وب‌اپلیکیشن اجرا می‌شوند. مهم‌ترین فولدر پروژه، فولدر app است که کدهای اجرایی اصلی پروژه در آن قرار می‌گیرد. به

شکل ۱۷: ساختار فایل در انگولار

<sup>۱</sup> Visual Studio

<sup>۲</sup> WebStorm

<sup>۳</sup> VS Code

طور پیش فرض مولفه app ساخته می شود که در واقع بزرگ ترین پدر همه ی مولفه های برنامه خواهد بود. فایل های با پسوند module.ts. مکان قرارگیری تعریف مولفه ها و واردسازی ماژول های دیگر است و همچنین اجازه استفاده بیرونی از مولفه های داخلی، در این قسمت قرار می گیرد. سه فایل با ساختار «نوع فایل.component.اسم مولفه» شکل کلی یک مولفه را تشکیل می دهد. فایل spec نیز فایل تست هر مولفه است.

## ۴-۲-۳- اتصال داده<sup>۱</sup> در انگولار

اتصال داده، مکانیزم هماهنگ کننده کد با چیزی که کاربر می بیند به خصوص مقادیر اطلاعاتی در برنامه کاربردی است. با این قابلیت می توان مقادیر را به راحتی در HTML قرار داد یا از آن حذف نمود. تعریف کردن اتصال داده به طور پیش فرض در انگولار بسیار راحت است. تنها کافی است مشخص کنیم که این متغیر یا رخ داد در مولفه ها به هم متصل شوند و چارچوب بقیه ی کارها را بر عهده می گیرد [۱۲].

انواع زیادی اتصال داده در انگولار قرار داده شده است که همه ی آن ها را می توان به سه دسته کلی تقسیم نمود [۱۲]:

- اتصال داده از source به view
- اتصال داده از view به source
- اتصال داده دو طرفه (از view به source و برعکس)

در جدول زیر به انواع مختلف اتصال داده و نحو آن ها پرداخته شده است [۱۲]:

جدول ۲۵: انواع اتصال داده در انگولار

دسته بندی	نحو	نوع
یک-طرفه از منبع اطلاعات به نمای هدف	<code>{{ expression }}</code> <code>[target]="expression"</code> <code>[ngClass]="expression"</code> <code>[ngStyle]="expression"</code>	Interpolation Property Attribute Class Style
یک-طرفه از نمای هدف به منبع اطلاعات	<code>(target)="statement"</code>	Event
دو-طرفه	<code>[(target)]="expression"</code>	Two-way

<sup>۱</sup> Data-Binding

برای مثال برای نوشتن اعمال یک کلاس به یک تگ HTML به طور شرطی به شکل زیر می‌نویسیم:

```
<mat-icon  
[ngClass]="element.role === 'CREATOR' ? 'fa-user-tie' : 'fa-user-tag'">  
</mat-icon>
```

شکل ۱۸: اضافه کلاس css به صورت شرطی به یک تگ

در کد بالا در صورتی که role مقدار «CREATOR» را داشته باشد کلاس CSS اولی و در غیر این صورت کلاس دوم به تگ «mat-icon» نسبت داده می‌شود. در صورتی که در حین اجرای برنامه کاربردی برای کاربر مقدار role عوض شود، مقدار کلاس نیز به‌روزرسانی خواهد شد.

## ۴-۲-۴ Lazy Loading در انگولار

یکی از مشکلات بزرگی که در برنامه‌های کاربردی تحت وب که از تکنولوژی SPA بهره می‌برند وجود دارد حجم زیاد بسته<sup>۱</sup> نهایی است. این مشکل زمانی خود را نشان خواهد داد که کاربر برای اولین بار سایت را بارگذاری می‌کند به طوری که باید زمان تقریباً زیادی را منتظر بماند تا صفحه‌ی معناداری از وبسایت مشاهده شود. راهکارهای متفاوتی برای حل این مشکل وجود دارد. یکی از راه‌های مهم آن شکستن قطعات کد به چندین بسته‌ی جدا از هم با تقسیم‌بندی بر اساس مکان استفاده است. این قابلیت به خاطر ماژولار بودن انگولار امکان‌پذیر است و می‌توان ماژول‌ها را بر حسب نیاز بارگذاری شوند. به این نوع بارگذاری به اصطلاح Lazy Loading گفته می‌شود.

به طور کلی در انگولار به دو شکل می‌توان از این قابلیت استفاده نمود:

- روش پیش‌فرض و خودکار آن جداسازی ماژول‌ها برای بارگذاری بر اساس مسیریابی و آدرس صورت می‌پذیرد. برای مثال وقتی کاربر وارد «domain.ir/b/blogname» می‌شود تنها ماژول اصلی و ماژول مخصوص وبلاگ بر روی سیستم کاربر بارگذاری می‌شود و دیگر ماژول‌ها مانند ماژول انجمن، بیهوده بارگذاری نخواهد شد.
- روش دیگر که روشی پیشرفته‌تر است به شکل دستی<sup>۲</sup> صورت می‌پذیرد. در این روش بر اساس نیاز و شرطی که برنامه‌نویس تعیین می‌کند ماژول‌های مورد نیاز

---

<sup>۱</sup> Bundle

<sup>۲</sup> Manually



را بارگذاری می‌کند. این روش نیازمند نوشتن منطق بارگذاری علاوه بر ماژول‌ها نیز می‌باشد.

در واقع انگولار روش پیش‌فرض که بیشترین استفاده را دارد به شکل آماده در اختیار توسعه‌دهندگان قرار داده است اما این انعطاف را نیز پیش‌بینی کرده است تا اگر برنامه‌نویس نیازمند منطق پیچیده‌تری نیز بود، بتواند از این قابلیت، به نحوی بهتر استفاده کند.

در این پروژه از این قابلیت پیشرفته برای بارگذاری ویجت‌ها استفاده شده است. به دلیل این‌که در ادامه‌ی پروژه و افزایش‌های آن ممکن است هزاران ویجت مختلف با توجه به نیاز کاربران نوشته شود و منطقی به نظر نمی‌رسد که همه‌ی این ویجت‌ها در یک ماژول واحد قرار بگیرند. راه حل این برنامه‌ی کاربردی به این صورت است که به ازای هر ویجت یک ماژول درست می‌شود. برای هر ویلاگ یا انجمن نیز لیست ویجت‌ها در دیتابیس نگهداری می‌شود و بر اساس آن‌ها به وسیله‌ی منطق بارگذاری نوشته شده، ماژول هر ویجت، بارگذاری شده تا به نمایش کاربر درآید.

## ۴-۲-۵- مدیریت وضعیت<sup>۱</sup>

به طور پیش‌فرض، در چارچوب انگولار از کتابخانه rxjs استفاده می‌شود. این کتابخانه برای برنامه‌نویسی رویدادگرا<sup>۲</sup> با استفاده از مشاهده‌گرها<sup>۳</sup> می‌باشد. این شیوه به آسان‌نمودن ساخت کدهای ناهمگام یا به اصطلاح callback-based کمک خواهد کرد.

این قابلیت در کنار سرویس‌های موجود به انگولار می‌تواند وضعیت‌های مختلف داده‌های برنامه را در خود نگه دارد و در صورت تغییر در آن‌ها به مشاهده‌گرهای آن داده اطلاع‌رسانی کند. همچنین با قرارگیری منطق‌های ارتباط بین فرانت و بک‌اند در این قسمت، تفکیک مناسبی بین این کدها و کدهای درون مولفه‌ای به وجود می‌آید و همچنین از چندباره تعریف کردن کدها با استفاده از الگوی سینگلتون و تزریق وابستگی جلوگیری می‌شود.

راهکار پیشرفته‌تری نیز برای مدیریت وضعیت وجود دارد که ساختار فرآیندی پیچیده‌تری را دارا است که این پیچیدگی در ابتدای کار و توسعه زمان‌بر است ولی در انتها به مدیریت وضعیت و دنبال کردن بهتر آن منجر خواهد شد و نگهداری آن را به مراتب آسان‌تر خواهد کرد. این معماری مدیریت وضعیت با نام «Redux» شناخته می‌شود که توسعه‌دهندگان ابتدایی آن مهندسین شرکت Facebook هستند. این معماری در انگولار در کتابخانه‌ای به نام

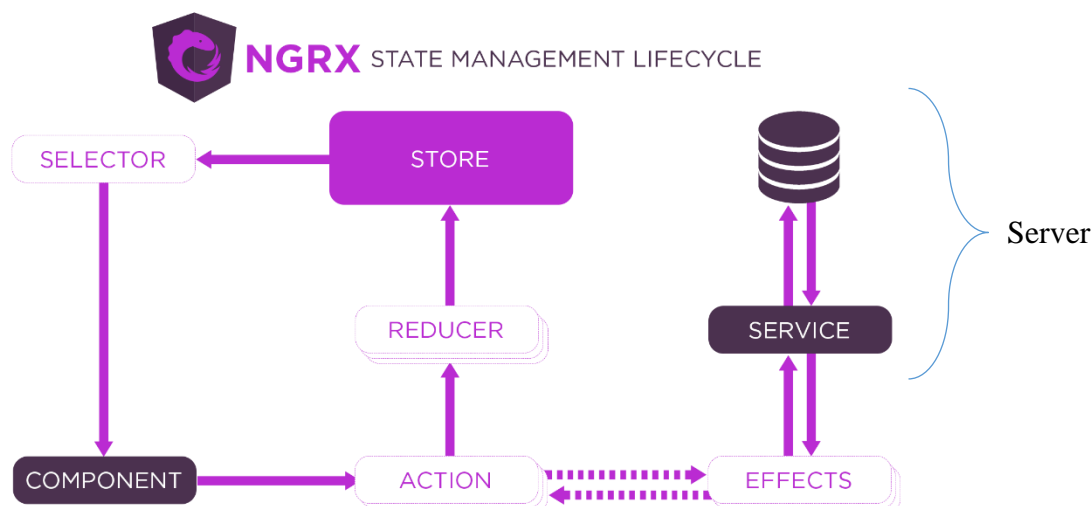
---

<sup>1</sup> State Management

<sup>2</sup> Reactive Programming

<sup>3</sup> Observable

«NgRx» پیاده‌سازی شده است. در شکل زیر معماری Redux در قالب انگولار به نمایش درآمده است.



شکل ۱۹: چرخه‌ی حیات معماری Redux در انگولار (کتابخانه NgRx)

در این معماری در قسمت انبار<sup>۱</sup> تمامی ساختمان داده‌ها با داده‌هایشان قرار می‌گیرد؛ با این هدف که از آن‌ها در کل برنامه، تنها یک منبع حقیقی<sup>۲</sup> وجود داشته باشد و مشاهده‌گرهای آن‌ها در صورت تغییر اطلاعات از این اتفاق با خبر شوند. مولفه‌ها برای دنبال کردن اطلاعات یک ساختمان داده نیز باید از انتخاب‌گر<sup>۳</sup>ها استفاده کنند.

برای این که بعد از انجام یک اتفاق، فعالیت را شروع کنیم یا بخواهیم اطلاعاتی که در مولفه‌های مختلف وجود دارد را تغییر دهیم، باید یک فعالیت<sup>۴</sup> را مخابره<sup>۵</sup> کنیم. بعد از این کار، NgRx در قسمت Reducer و Effects به دنبال این فعالیت می‌گردد و در صورت وجود چنین فعالیتی در آن‌ها، به اجرا آن می‌پردازد. همان‌طور که در تصویر نیز مشخص است Effects مسئول ارتباط با سرور یا به طور کلی‌تر مسئول فعالیت‌های ناهمگام است. Reducer نیز مسئول انجام فعالیت‌های همگام و به‌روزرسانی داده‌های موجود در انبار است.

این ساختار، نگاهی فرآیندی به مدیریت داده‌ها دارد و هر جزء مسئول فعالیتی به‌خصوص است تا از تغییرات ناخواسته جلوگیری شود. برای مثال از تغییر اطلاعات داخل انبار به غیر از فرایند گفته شده جلوگیری می‌شود و توسعه‌دهنده اگر ناخواسته این کار را انجام بدهد با خطای زمان اجرا مواجه خواهد شد.

<sup>۱</sup> Store

<sup>۲</sup> Single Source of Truth

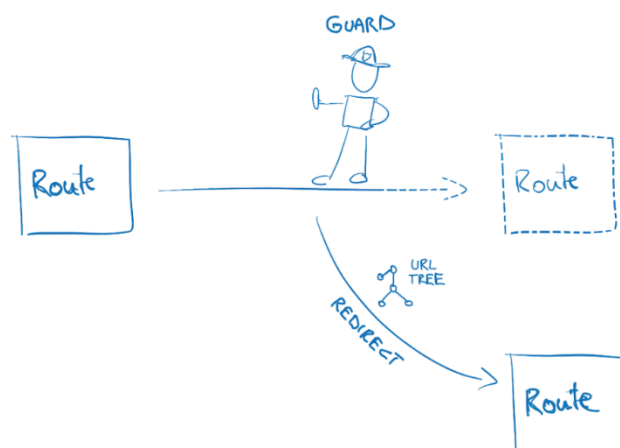
<sup>۳</sup> Selector

<sup>۴</sup> Action

<sup>۵</sup> Dispatch

## ۴-۲-۶- محافظ‌ها در انگولار

یکی از قابلیت‌های ویژه در چارچوب انگولار، محافظ‌های مسیریابی است. این محافظ‌ها بر اساس منطقی که در داخلشان قرار می‌گیرد از ورود افراد نامعتبر به صفحات جلوگیری می‌کند. برای مثال افرادی که وارد نشده‌اند، مجاز به دسترسی به صفحه‌ی پروفایل نیستند یا افرادی که وارد شده‌اند، نباید صفحه‌ی ثبت‌نام را مشاهده کنند.



شکل ۲۰: محافظ در مسیریابی

## ۴-۳- چارچوب نست‌جی‌اس

در این قسمت برخی از مفاهیم چارچوب نست، مانند ساخت یه برنامه‌ی ساده، ساختار پروژه‌ها و برنامه‌نویسی متا شرح داده خواهد شد.

### ۴-۳-۱- استفاده از چارچوب نست‌جی‌اس

برای نصب این چارچوب همانند انگولار از yarn استفاده می‌شود:

```
> yarn add @nestjs/cli
```

بعد از نصب این چارچوب با استفاده از دستور زیر پروژه‌ی جدیدی با نام «community» و با مدیریت پکیج yarn را ایجاد می‌کنیم:

```
> nest new community --package-manager=yarn
```

---

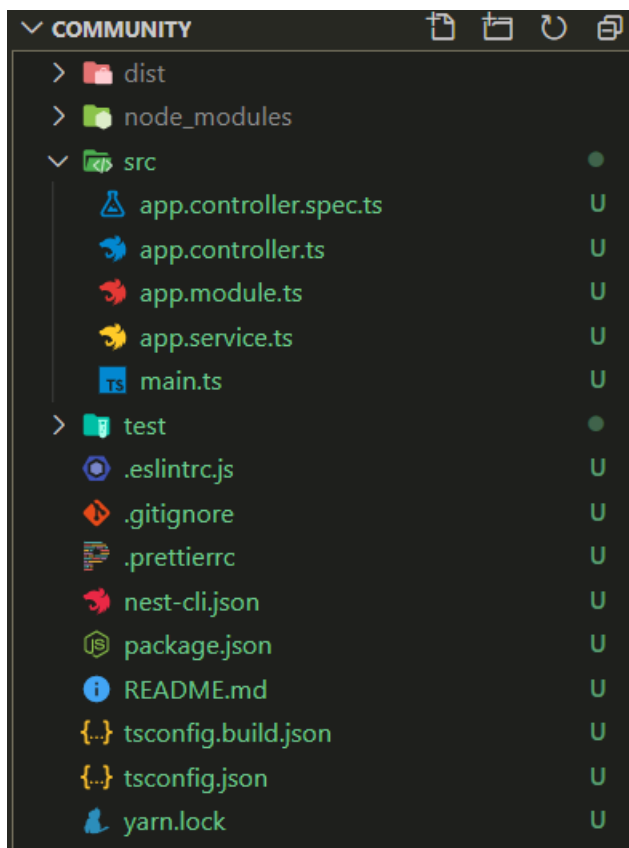
<sup>1</sup> Guard

برای اجرا اولین برنامه، دستور زیر را اجرا می‌کنیم:

با اجرا این دستور سرور در حالت توسعه اجرا خواهد شد و پورت پیش‌فرض آن ۳۰۰۰ می‌باشد.

```
> yarn run start:dev
```

## ۴-۳-۲- ساختار پروژه در نست‌جی‌اس



شکل ۲۱: ساختار پروژه در نست‌جی‌اس

ساختار پروژه‌ها در نست شباهت

زیادی به انگولار دارد. همانطور که در تصور مقابل نیز مشخص است سه پوشه `node_modules`، `test` و `src` در ساختار هر دو پروژه وجود دارد. به جای فایل `angular.json` نیز فایل `nest-cli.json` وجود دارد. اولین کدی که از پروژه اجرا خواهد شد، فایل `main.ts` در پوشه `src` است. در این فایل تنظیمات کلی نست قرار خواهد گرفت. برای مثال تنظیمات پورت، انتخاب چارچوب زیرین (اکسپرس یا فستیفای)، استفاده از افزونه‌های آن‌ها و پیکربندی‌های امنیتی می‌توان از مهم‌ترین این تنظیمات باشد. در `app.module` نیز که بزرگ‌ترین پدر همه‌ی ماژول‌ها خواهد بود علاوه بر تعریف زیرماژول‌ها، دیتابیس فراخوانی شده و آماده‌ی استفاده می‌شود.

به جای مولفه‌ها در نست‌جی‌اس کنترلرها

در کنار سرویس‌ها قرار می‌گیرند. کنترلرها وظیفه‌ی گوش‌فرادادن به درخواست‌ها در حوزه‌ی مربوط به خود را دارند. به طور دقیق‌تر چارچوب نست بعد از بررسی درخواست گرفته‌شده از کاربر به تفسیر مسیر درخواست می‌پردازد و بر اساس آن کنترلر مربوط به آن را صدا می‌زند. کنترلر مربوط نیز به بررسی وجود زیرمجموعه‌های خود پرداخته و در صورت مطابقت با یکی از الگوهای زیرمجموعه‌ی خود، تابع زنجیرشده به آن را صدا می‌زند. کنترلرها پس از کنترل دسترسی و تبدیل اطلاعات به داده‌های تمیز، سرویس‌های مربوط به خودشان را صدا زده تا عملیات‌های منطقی و ارتباط با پایگاه‌داده توسط آن‌ها انجام شود. البته کنترلرها، کنترل دسترسی و تبدیل اطلاعات را به کمک اجزای دیگر انجام می‌دهند. این امر با ساختار نحوی «Meta-Programming» به زیبایی میسر می‌شود.

## ۴-۴ - Meta-Programming در نست جی اس

یکی از ویژگی‌های تایپاسکرپت که به میزان بسیار زیاد در چاقوب نست به چشم می‌آید، استفاده از دکوراتورها است. برای مثال در تصویر ۲۱ تابع «deleteBlog» با سه دکوراتور پیش‌از اجرا و یک دکوراتور در ورودی‌اش دیده می‌شود. دکوراتور @Delete متد درخواست HTTP و الگوی زیرساختاری در حوزه‌ی «/blog/» را نشان‌دهد. دکوراتور @UseGuards مسئول اجرای محافظ‌های تعریف‌شده در ورودی خودش است که در اینجا محافظ اول به بررسی صحت اطلاعات کاربری فرد در توکن و محافظ دوم به اعتبارسنجی عضویت کاربر در بلاگ می‌پردازد. دکوراتور آخر یک کمک‌کننده<sup>۱</sup> به SocialGuard است. در واقع اگر این دکوراتور وجود داشته باشد، SocialGuard علاوه بر کار قبل به بررسی نقش کاربر

```
@Roles([SocialUserRole.CREATOR])
@UseGuards(AuthGuard(), SocialGuard)
@Delete('/:sid')
deleteBlog(@Param(ValidationPipe) socialParams: SocialParams) {
  return this.blogService.deleteBlog(socialParams.sid);
}
```

شکل ۲۲: استفاده دکوراتورها و پایپ در قسمتی از یک کنترلر

عضو شده در بلاگ می‌پردازد و تنها در صورتی که نقش کاربر در بلاگ، «سازنده» باشد اجازه اجرا تابع حذف بلاگ داده خواهد شد.

دکوراتور @Param که در ورودی تابع قرار دارد، مسئولیت استخراج پارامترها (در اینجا sid) است. ورودی داخل آن یعنی ValidationPipe به کمک کتابخانه‌ی جانبی «class-validator» به بررسی صحت ساختاری ساختمان داده و مقدار نوع‌داده‌ی داخل پارامترها می‌پردازد و در صورت وجود مشکل، به صورت خودکار پیغام خطا را به فرانت‌اند برمی‌گرداند.

بعد از همه‌ی این مراحل، در صورتی که هیچ خطایی رخ ندهد، از سرویس «blogService» تابع مربوط به پاک‌کردن بلاگ صدا زده می‌شود و نتیجه نهایی به کاربر برگردانده خواهد شد.

## ۴-۵ - جمع‌بندی

در این فصل به شرح ساختار درختی پروژه و معماری آن و فعالیت‌های مهم در قسمت پیاده‌سازی پرداخته شد. همچنین چالش‌های اصلی که در طول بخش پیاده‌سازی به وجود آمد، شرح داده شد و راهکارهای آن مورد تحلیل قرار گرفت و در نهایت سعی شد بهترین روش برای حل آن چالش استفاده شود.

---

<sup>۱</sup> Helper

## فصل پنجم

### تست و آزمون نرم افزار

#### ۵-۱- مقدمه

در فرایند توسعه‌ی نرم افزار، یکی از مرحله‌های مهم آزمون و تست نرم افزار است. این فعالیت به طور کلی انجام می‌گیرد تا مطمئن شویم: [۱۱]

- آیا نیازمندی‌های اولیه برآورده شده است؟
- هر قسمت به درستی عمل می‌کنند؟
- نرم افزار قابلیت نگهداری خوبی دارد؟
- به شکل خوبی برنامه‌نویسی شده است؟

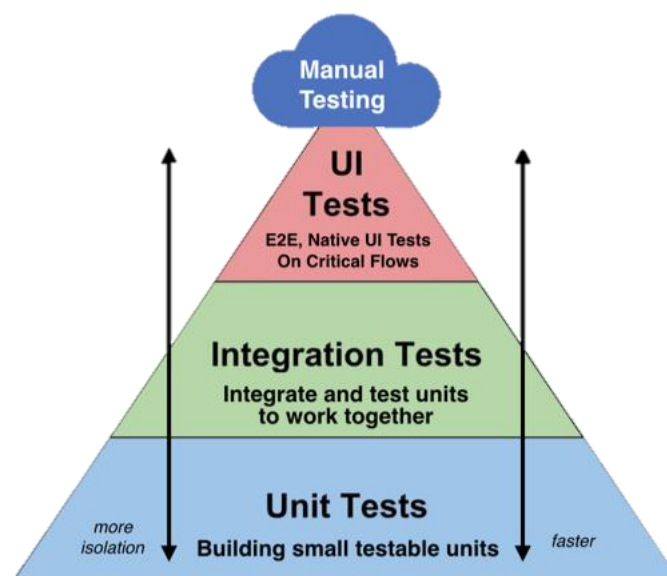
برای پاسخ به تمام این سوالات آزمون‌های مختلفی وجود دارد که در این پروژه از ساختار شکل ۲۳ استفاده شده است.

در این هرم که آزمون برنامه‌نویسی خواهد شد، ابتدا توسط مایک کوهن طراحی شده است. در آغاز، آزمون واحدها<sup>۱</sup> که در این پروژه برای مثال هر توابع مولفه، سرویس، محافظ و کنترلر، به شکل مجزا تست می‌شود تا اطمینان حاصل شود که توابع و ساختارها به شکل جزئی و منفرد به درستی عمل می‌کنند.[۱۱]

در مرحله‌ی بعد، آزمون سطح بالاتر هرم یعنی تست یکپارچه‌سازی قرار دارد. در این مرحله از تعامل بخش‌های کوچک اطمینان می‌یابیم و بررسی می‌کنیم که آیا واحدهای کوچک‌تر می‌توانند با یکدیگر کار به درستی کار کنند.

تست دیگر که راس هرم را تشکیل می‌دهد تست واسط کاربری است که با نام‌های آزمون E2E<sup>۲</sup> و یا آزمون مرورگر شناخته می‌شود. این آزمون برای بررسی صحت کارکرد کل سیستم استفاده می‌شود و نوشتن آزمون برای آن‌ها سخت و زمان‌بر است.

علاوه بر این‌ها آزمون قابلیت استفاده<sup>۳</sup> واسط کاربری انجام شده است که در انتها آمده است.



## ۵-۲- تست واحد

همان‌طور که در بخش قبل گفته شد، تست پایه و ابتدایی برای بررسی صحت هر واحدی چه در بخش فرانت‌اند و چه در بخش بک‌اند تست واحد خواهد بود. ابزار تست برای بخش فرانت‌اند برای این سطح Jasmine و از اجراکننده‌ی آزمون Karma است که توسط خود انگولار نیز پیشنهاد شده است. در بخش بک‌اند از کتابخانه‌ی Jest استفاده می‌شود که توسط شرکت فیس‌بک توسعه داده شده است و پیشنهاد چارچوب نست‌جی‌اس می‌باشد. این کتابخانه‌ها شباهت زیادی به هم دارند و می‌توان گفت Jest چندین لایه روی کتابخانه‌ی Jasmine را برای برنامه‌نویسان فراهم می‌کند.

<sup>۱</sup> Unit Testing

<sup>۲</sup> End-to-End

<sup>۳</sup> Usability

## ۵-۲-۱- تست واحد در فرانت‌اند

در پروژه برای مثال برای تست مولفه‌ی App که آیا مولفه به درستی ساخته می‌شود و آیا بعد از اینکه مولفه به حالت پایدار رسید، نشانه‌ی بارگذاری بر روی صفحه نباشد آزمون شکل ۲۳ را می‌نویسیم. روند کلی نوشتن ساختار یک آزمون واحد به شرح زیر است:

- توضیح کلی در مورد آزمون (describe)
- قبل از هر آزمون چه کارهایی انجام شود (beforeEach)
- در beforeEach ماژول‌ها مورد نیاز را یا وارد می‌کنیم و یا رفتار آن‌ها را تقلید<sup>۱</sup> می‌کنیم.
- مولفه را می‌سازیم.
- آزمون‌ها را می‌نویسیم.

همانطور که در تصویر ۲۴ مشخص است، در خط اول متنی که توضیح‌دهنده‌ی این بخش از آزمون است، نوشته شده است. سپس در قسمت beforeEach برای هر آزمون که با «it» مشخص می‌شود پیکربندی

```
describe('AppComponent', () => {
  let fixture: ComponentFixture<AppComponent>;
  beforeEach(async(() => {
    TestBed.configureTestingModule({
      imports: [RouterTestingModule],
      declarations: [AppComponent],
      providers: [provideMockStore()]
    }).compileComponents();
    fixture = TestBed.createComponent(AppComponent);
    fixture.detectChanges();
  }));
  it('should create the app', () => {
    const app = fixture.debugElement.componentInstance;
    expect(app).toBeTruthy();
  });
  it('shouldn\'t be in loading after rendering', async () => {
    const compiled = fixture.debugElement.nativeElement;
    await fixture.whenRenderingDone();
    expect(compiled.querySelector('#loading')).toBeFalsy();
  });
});
```

شکل ۲۴: آزمون واحد برای مولفه app

<sup>۱</sup> Mock



ماژول انجام شده است. برای مثال مدیریت وضعیت واقعی استفاده نشده و از داده‌ی تقلیدی استفاده شده است. در انتهای آن نیز مولفه ساخته و ارائه شده است.

سپس آزمون اول با توضیح مربوط به آن آمده است و تابع expect و ورودی‌ها بررسی نموده‌ایم که آیا مولفه app ساخته شده است یا خیر.

در آزمون دوم نیز منتظر انجام کامل ترجمه و ارائه شده‌ایم و بر اساس آی‌دی نشانگر بارگذاری از وجود نداشتن آن بر روی صفحه مطمئن شده‌ایم.

مثال بعد آزمون محافظ Auth در قسمت فرانت‌اند است که در اینجا مسیریاب تقلید شده است. دلیل این امر این است که در تست واحد، ما به درستی ماژول مسیریاب کاری نداریم و صحت خود محافظ برایمان مهم است. در این قسمت نیز دو آزمون وجود دارد.

آزمون اول بررسی می‌کند اگر کاربر لاگین کرده است و توکنی وجود دارد محافظ اجازه‌ی عبور دهد و آزمون دوم بررسی می‌کند که اگر توکنی وجود ندارد کاربر اجازه‌ی عبور نداشته باشد و همچنین بررسی می‌کنیم که متد تغییر مسیر و رفتن به صفحه خطا انجام شود. در شکل ۲۵ کد این آزمون آمده است.

```
const dummyJWT = 'Dummy Valid JWT';
class MockRouter {
  navigateByUrl(path) { }
}
describe('AuthGuard', () => {
  describe('canActive', () => {
    let authGuard: AuthGuard;
    let router: any;
    beforeEach(() => {
      router = new MockRouter();
      authGuard = new AuthGuard(router);
    });
    it('should return true for a logged in user', () => {
      localStorage.setItem(ACCESS_TOKEN_KEY, dummyJWT);
      expect(authGuard.canActivate(undefined, undefined)).toEqual(true);
    });
    it('should navigate to 404 for a logged-out user', () => {
      localStorage.removeItem(ACCESS_TOKEN_KEY);
      spyOn(router, 'navigateByUrl');
      expect(authGuard.canActivate(undefined, undefined)).toEqual(false);
      expect(router.navigateByUrl).toHaveBeenCalledWith('/error/404');
      localStorage.setItem(ACCESS_TOKEN_KEY, dummyJWT);
    });
  });
});
```

شکل ۲۵: آزمون واحد برای محافظ Auth

آزمون واحد دیگری که در بخش فرانت‌اند انجام شده است، آزمون لوله «PersianDate» است. در این آزمون انتظار داریم در ازای ورودی‌های مشخصی که خودمان می‌دهیم خروجی مورد انتظار را دریافت کنیم که مانند شکل ۲۶ می‌نویسیم.

```
describe('Persian Date Pipe', () => {
  let persianDatePipe: PersianDatePipe;
  beforeEach(() => {
    persianDatePipe = new PersianDatePipe();
  });
  it('should convert 1997/07/27 to ۱۳۷۶/۵/۵', () => {
    const date = '1997/07/27';
    const resultMustBe = '۱۳۷۶/۵/۵';
    const result = persianDatePipe.transform(date);
    expect(result).toBe(resultMustBe);
  });
  it('should convert 2002/05/15 to ۱۳۸۱/۲/۲۵', () => {
    const date = '2002/05/15';
    const resultMustBe = '۱۳۸۱/۲/۲۵';
    const result = persianDatePipe.transform(date);
    expect(result).toBe(resultMustBe);
  });
});
```

شکل ۲۶: آزمون لوله تبدیل تاریخ شمسی

مورد بعد تست مدیریت وضعیت است. در آزمون کاهنده<sup>۱</sup>، ابتدا بررسی می‌شود که مقدار پیش‌فرض برابر با وضعیت داده‌ی ابتدایی باشد که انتظار داریم و مشخص کرده‌ایم. در آزمون بعد انتظار داریم، زمانی که اجتماعی (Social) واکنشی می‌شود و قرار است در انباره ذخیره شود، به شکل مورد انتظار این اتفاق روی دهد. آزمون‌های ذکر شده در شکل ۲۷ آماده است.

آزمون بعد، بررسی عملکرد «effect» در اجتماع است که در شکل ۲۸ آمده است. برای نوشتن این آزمون، مسیریاب و HttpClient را که وظیفه فرستادن درخواست به سمت سرور را بر عهده دارد تقلید می‌کنیم. همچنین کاهنده بخش اجتماع را نیز تقلید کرده‌ایم و به شکل دستی effect مربوط را فرا می‌خوانیم. فرق دیگری که این تست با تست‌های قبل دارد این است که توابع effect مقداری از نوع Observable یا مشاهده‌گر برمی‌گردانند. در حالت عادی کتابخانه‌های تست به دلیل اینکه این اتفاق در لحظه نمی‌افتد از روی آن رد می‌شوند و باید کمی تغییر در ساختار آزمون ایجاد کرد. در این قسمت، از ورودی تابع callback

---

<sup>۱</sup> Reducer

که ورودی تابع «it» است استفاده می‌کنیم. به این صورت که در آن effect به اصطلاح subscribe می‌شود و در آخرین خط داخل آن، آن ورودی داخل callback فراخوانی می‌شود.

```
describe('Store [Social] | Reducer', () => {
  const INIT_STATE = { ...fromReducer.INIT_STATE };
  it('Should return the default state', () => {
    const initialState = { ...INIT_STATE };
    const state = fromReducer.reducer(undefined, { type: null });
    expect(state).toEqual(initialState);
  });
  it('Should reduce fetched Social', () => {
    const initialState = { ...INIT_STATE };
    const payload = {social: [{ name: 'testSocial' }]};
    const action = fromActions.SocialFetched(payload);
    const state = fromReducer.reducer(initialState, action);
    expect(state.social).toEqual(payload.social);
  });
});
```

شکل ۲۸: آزمون واحد Social Reducer

```
describe('Store [Social] | Effects', () => {
  let actions$: Observable<Action>; let effects: SocialEffects;
  const httpClientSpy = jasmine.createSpyObj('HttpClient', ['post']);
  const routerSpy = jasmine.createSpyObj('Router', ['navigateByUrl']);
  beforeEach(() => {
    const moduleRef = TestBed.configureTestingModule({
      imports: [NoopAnimationsModule, MatSnackBarModule],
      providers: [provideMockActions(() => actions$), SocialEffects, provideMockStore(), { provide: HttpClient, useValue: httpClientSpy }, { provide: Router, useValue: routerSpy}]);
    effects = moduleRef.inject(SocialEffects);
  });
  it('should listen to Create Social and dispatch User Social Created', (done) => {
    httpClientSpy.post = () => of(1);
    actions$ = of({});
    effects.socialCreate.subscribe(action => {
      expect(action).toEqual({
        type: '[User] User Social Created',
        social: {}
      });
      done();
    });
  });
});
```

شکل ۲۷: آزمون واحد Social Effects به طور خلاصه

## ۵-۲-۲- تست واحد در بخش بک‌اند

در قسمت بک‌اند که از کتابخانه‌ی Jest استفاده شده است به بررسی واحد کنترلر search می‌پردازیم. در این آزمون بررسی می‌کنیم که آیا ورودی‌هایی که انتظار داریم به عنوان مورد آزمون به تابع «getHomepagePostData» داده شود، مقادیر مورد انتظار را به درستی می‌گیرند و نوع مورد انتظار را برمی‌گردانند.

```
describe('SearchController', () => {
  let searchController: SearchController;
  let searchService: SearchService;
  function mockModel(dto: any) {
    this.data = dto;
    this.save = () => this.data;
  };
}
beforeEach(async () => {
  const moduleRef = await Test.createTestingModule({
    controllers: [SearchController],
    providers: [SearchService, AppLogger,
      { provide: getModelToken('Post'), useValue: mockModel },
      { provide: getModelToken('Social'), useValue: mockModel },
      { provide: getModelToken('User'), useValue: mockModel }],
  }).overrideGuard(AuthGuard()).useValue({ canActivate: () => true }).compile();
  searchService = await moduleRef.resolve(SearchService);
  searchController = moduleRef.get<SearchController>(SearchController);
});
it('should get posts from getHomepagePostData from service and return it', async ()
=> {
  const result = { length: 2, posts: [{ } as Post, { } as Post] };
  jest.spyOn(searchService, 'getHomepagePostData')
    .mockImplementation(() => Promise.resolve(result));
  expect(await searchController.getHomepagePostData(
    { page: '1', itemsPerPage: '10', sortBy: PostSortBy.NEWEST }, undefined))
    .toBe(result);
});
});
```

شکل ۲۹: آزمون کنترلر search

## ۵-۳- آزمون یکپارچه‌سازی

آزمون یکپارچه‌سازی به بررسی و صحت کارکرد تعامل چندین ماژول در کنار هم می‌پردازد. در اینجا به بررسی صحت عملکرد ورود کاربران عضو به سایت پرداخته خواهد شد. ماژول‌ها و مولفه‌ای که ارتباط آن‌ها با هم بررسی می‌شود، مولفه «Login»، مدیریت وضعیت «Auth» و ماژول‌هایی مثل «ReactiveForms» و «MatDialog» است.

```
describe('Login Test', () => {
  ... // some declaration was here
  beforeEach(async () => {
    await TestBed.configureTestingModule({
      declarations: [LoginComponent],
      imports: [
        ... // some imports were here
        StoreModule.forRoot(rootReducer),
        EffectsModule.forRoot([AuthEffects, UserInfoEffects])
      ],
      providers: [
        { provide: MAT_DIALOG_DATA, useValue: {} },
        { provide: MatDialogRef, useValue: dialogMock },
        { provide: HttpClient, useValue: httpClientSpy },
      ]
    }).compileComponents();
    ... // create component and inject store and set value in form were here
  });
  it('should create the component', () => expect(component).toBeTruthy());
  it('should login successfully', async(() => {
    httpClientSpy.post = () => of({ accessToken: dummyJWT });
    let loginSuccess = false;
    spyOn(component, 'onClose').and.callFake(() => { // when user logged in successfully, dialog will close. so spyOn it.
      loginSuccess = true;
    });
    clickByCSS('#loginClick');
    fixture.detectChanges(); // render page
    expect(loginSuccess).toBe(true);
    expect(getLoadingCSS()).not.toBeTruthy(); // not to be in loading mode
  }));
  it('should show wrong password/username ERROR', () => {
    httpClientSpy.post = () => throwError({ status: 401 });
    clickByCSS('#loginClick');
    fixture.detectChanges();
    expect(getErrorText()).toBeTruthy();
    expect(getLoadingCSS()).not.toBeTruthy();
  });
});
```

شکل ۳۰: آزمون یکپارچه‌سازی انجام لاگین در سایت

```
DevTools listening on  
ws://127.0.0.1:62935/devtools/browser/030d81a3-  
2cfc-4531-aa3c-4ffa260d05c4
```

Jasmine started

### Community App

#### Homepage

- ✓ should open base url
- ✓ should HomeComponent exist
- ✓ should open login dialog
- ✓ should click on register button and navigate

#### Register Page

- ✓ should appear title
- ✓ should fill the username
- ✓ should fill the email
- ✓ should fill the password
- ✓ should fill the passwordConfirm
- ✓ should check the agreement
- ✓ should click on register button
- ✓ should already logged in

#### Create a Blog

- ✓ should navigate to create blog page
- ✓ should appear title
- ✓ should fill social name
- ✓ should fill social title
- ✓ should fill social description
- ✓ should select the subject
- ✓ should click on create and navigate to blog page
- ✓ should show title of the blog with blogTest63893

#### Create Post

- ✓ should click on create post and navigate
- ✓ should fill post title
- ✓ should fill post subtitle
- ✓ should fill post text
- ✓ click on publish post and navigate to blog main page

#### Logout

- ✓ should logout from webapp

Executed 26 of 26 specs SUCCESS in 20 secs.

## ۴-۵- آزمون رابط کاربری یا E2E

در آزمون E2E یا رابط کاربری سعی می‌شود سناریوهایی طراحی شود که کاربر انجام می‌دهد و افعال وی شبیه‌سازی می‌شود [۱۰]. به طور مثال کاربر برای ورود به سایت بر روی منو کلیک کرده و بعد روی دکمه‌ی «ورود/ثبت‌نام» کلیک می‌کند. در این آزمون نیز با استفاده از شبیه‌ساز، کارها و سناریوهایی که ممکن است انجام دهد مانند کلیک کردن، نوشتن و ... شبیه‌سازی می‌شود.

در آزمونی که نتیجه‌ی آن در شکل ۳۰ به نمایش درآمده است چنین سناریوی در نظر گرفته شده است:

۱. کاربر وارد صفحه‌ی اصلی می‌شود.
۲. کاربر ثبت‌نام می‌کند که بعد از آن، به طور پیش‌فرض لاگین شده است.
۳. کاربر بلاگ جدیدی می‌سازد.
۴. کاربر در بلاگ ساخته شده پست جدیدی ایجاد می‌کند.
۵. کاربر از حساب کاربری خود خارج می‌شود.

این آزمون با استفاده از ابزارهایی نظیر «ChromeDriver» و چارچوب آزمونی که برای انگولار طراحی شده است یعنی «Protractor» نوشته شده است.

شکل ۳۱: نتیجه آزمون E2E انجام شده که همه‌ی آن با موفقیت گذرانده شده است.

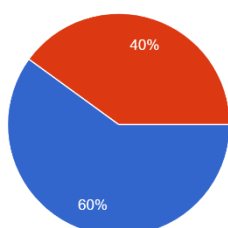
## ۵-۵- آزمون واسط کاربری

در این آزمون به ۵ اصل مهم قابلیت استفاده یعنی رضایت‌مندی، قابلیت یادگیری، قابلیت رویت، کارایی و کنترل خطا و کاربر پرداخته می‌شود.

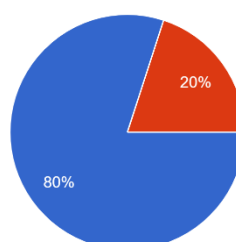
برای بررسی این موضوع، پرسش‌نامه‌ای تدارک دیده شده است. کاربران بعد از مقداری کار با این برنامه پرسش‌نامه را پر کرده‌اند و نظرات خود را در مورد این معیارها با سوالات طرح شده مشخص ساخته‌اند.

### • رضایت‌مندی

تجربه‌ی خود از این برنامه را تا چه حد لذتبخش می‌دانید؟  
5 responses

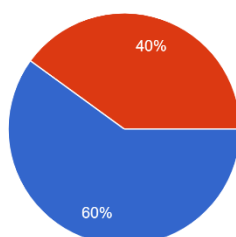


تا چه حد عملکرد این برنامه رو قابل قبول می‌دانید؟  
5 responses



● عالی  
● بسیار خوب  
● متوسط  
● ضعیف  
● خیلی ضعیف

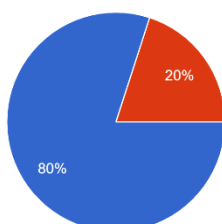
تا چه حد این برنامه نیازهای مشخص‌شده را برطرف می‌سازد؟  
5 responses



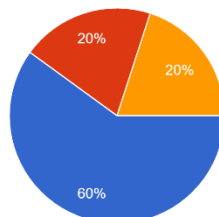
شکل ۳۲: نظرسنجی واسط کاربری (رضایت‌مندی)

### • قابلیت یادگیری

تا چه حد اثرات فرایندها در برنامه قابل رویت است؟  
5 responses

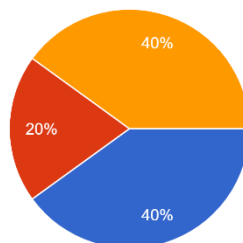


تا چه حد این وب اپلیکیشن از استانداردهایی که شما در طراحی وب می‌شناسید تبعیت می‌کند؟  
5 responses



● عالی  
● بسیار خوب  
● متوسط  
● ضعیف  
● خیلی ضعیف

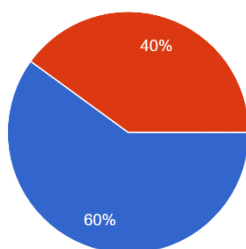
تا چه حد کشف قابلیت‌های برنامه را آسان می‌دانید؟  
5 responses



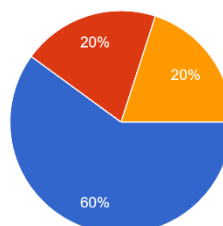
شکل ۳۳: نظرسنجی واسط کاربری (قابلیت یادگیری)

## • قابلیت رویت

تا چه حد قابلیت‌های برنامه در توجه شما قرار دارند؟  
5 responses

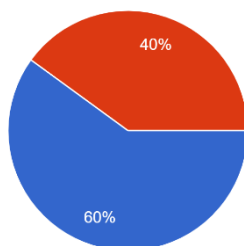


تا چه حد کاربر از جابجایی‌ها و محل فعلی خود آگاه است؟  
5 responses



عالی  
بسیار خوب  
متوسط  
ضعیف  
خیلی ضعیف

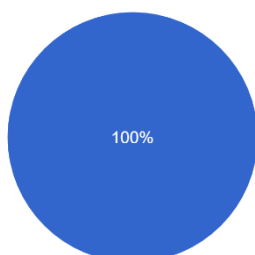
تا چه حد بلافاصله از اثرات یک فرآیند آگاه می‌شوید؟  
5 responses



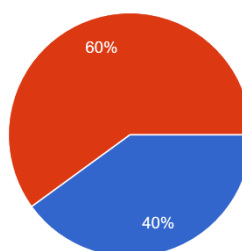
شکل ۳۴: نظرسنجی واسط کاربری (قابلیت رویت)

## • کارایی

سرعت این برنامه را چگونه ارزیابی می‌کنید؟  
5 responses



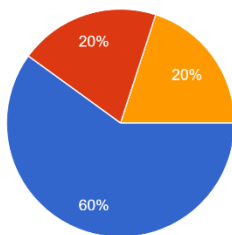
تا چه حد عملیات‌های این برنامه قابل حفظ کردن هستند تا در استفاده‌های بعدی عملیات‌های کاربر آسان تر شود؟  
5 responses



عالی  
بسیار خوب  
متوسط  
ضعیف  
خیلی ضعیف



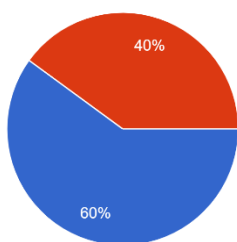
دسترسی با موارد مرتبط با یکدیگر را در این برنامه چگونه ارزیابی می‌کنید؟  
5 responses



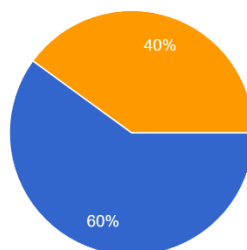
شکل ۳۵: نظرسنجی واسط کاربری (کارایی)

## • کنترل خطا و کاربر

تا چه حد برنامه از خطاهای کاربر جلوگیری می‌کند؟  
5 responses

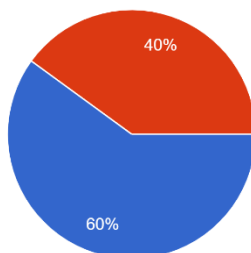


تا چه حد پیام‌ها در این برنامه کاربر را به راه‌نمایی می‌کند؟  
5 responses



● عالی  
● بسیار خوب  
● متوسط  
● ضعیف  
● خیلی ضعیف

تا چه حد پیام‌ها با نتایج آن‌ها در این برنامه هم‌تراز هستند؟  
5 responses



شکل ۳۶: نظرسنجی واسط کاربری (کنترل خطا و کاربر)

همانطور که قابل مشاهده است اکثر کاربران با نسبت خوبی از برنامه راضی هستند و تنها در بخش کشف قابلیت‌ها و نمایش پیام‌ها در طراحی واسط ابهاماتی برای کاربر وجود دارد. برای کشف قابلیت علاوه بر افزایش مشاهده‌پذیری قابلیت‌ها می‌توان از راهنمای گام‌به‌گام<sup>۱</sup> استفاده نمود. همچنین برای نمایش پیام‌ها می‌توان، توضیحات خطا را کامل‌تر کرد و راهکاری اندیشید تا در حین حل مشکل به کاربران راهنمایی بدهد.

<sup>۱</sup> Walkthrough Guide

## ۵-۶- جمع‌بندی

در این فصل به روش‌های آزمون، توضیحات آن و مثال‌های واقعی انجام شده در پروژه‌ی جاری پرداخته شد و درباره‌ی ابزارهای آن نیز توضیح مختصری داده شد. نوشتن آزمون و فلسفه‌ی آن به نوعی، ساختار متفاوتی با برنامه‌نویسی و توسعه عادی دارد و نوشتن سناریو صحیح و پیاده‌سازی آن، کاری بسیار زمان‌بر و نیازمند مطالعه بیشتر است. نوشتن آزمون برای اکثر پروژه‌ها، به دلیل این که نیازمند بروزرسانی و نگهداری هستند به شکلی بدیهی لازم است زیرا در بلندمدت، زمان اولیه‌ای که به نوشتن و طرح آزمون‌ها گذاشته می‌شود جبران و همچنین تمام جوانب کار و خطاهای ممکن بررسی و مرتفع خواهد شد.

## فصل ششم

### جمع‌بندی و پیشنهادها

#### ۶-۱ - جمع‌بندی و دورنما

در انجام این پروژه تلاش شد تا با در نظرگیری تمامی جنبه‌های تاثیرگذار بر آن، از بهترین و کاربردی‌ترین روش‌ها در توسعه‌ی برنامه‌ی کاربردی مربوطه استفاده شود.

همچنین در کنار محصول نرم‌افزاری تولیدشده به ارائه‌ی مستندات جامع و دقیق پرداخته شد تا جنبه‌های مختلف این پروژه را پوشش دهد و درآینده به‌سادگی مورد استفاده قرار گیرد.

این پروژه پس از تکمیل و در صورت پیداشدن سرمایه‌گذار، مورد استفاده تمام افراد جامعه قرار خواهد گرفت و تجربه‌ای جدید در طراحی و ساخت انجمن و بلاگ را برای آن‌ها رقم خواهد زد.

همچنین این پروژه می‌تواند مقدمه‌ای برای توسعه‌ی هرچه بیشتر برنامه‌های کاربردی و شبکه‌های اجتماعی، با تاکید بر احترام هرچه بیشتر به نظر و سلیقه‌ی کاربران و با انعطاف‌پذیری بالا باشد.

## ۶-۲- پیشنهادها

باتوجه به اینکه این پروژه نمونه‌ای ابتدایی از یک محصول نهایی است، در زمینه‌های گوناگون به ارتقا و حتی اعمال تغییرات جزئی نیازمند است.

در ادامه به ذکر چند مورد از پیشنهادهای مطرح‌شده درباره‌ی این پروژه خواهیم پرداخت:

- سیستم جستجوی داخلی قوی
- پیام خصوصی به افراد
- افزودن قالب‌هایی مانند فروشگاه و صفحات مخصوص محتوای صوتی
- افزودن هشتگ<sup>۱</sup> (#) و نام‌بردن<sup>۲</sup> (@)
- سیستم انعام‌دهی<sup>۳</sup> بر پایه‌ی ارزش با پشتوانه‌ی واقعی بر روی بستر بلاکچین<sup>۴</sup>
- مرتبط‌سازی معنادارتر بلاگ و انجمن به یکدیگر
- رابط کاربری زیباتر و با جزئیات و انعطاف بیشتر

---

<sup>۱</sup> Hashtag

<sup>۲</sup> Mention

<sup>۳</sup> Tip

<sup>۴</sup> Blockchain

- [۱] Invader Bethany, "What Is Really On The Deep Web,?" 2 May 2017 [Online]. Available: <https://www.theodysseyonline.com/deep-web> [Accessed 12 05 2020].
- [۲] Typed JavaScript at Any Scale. [Online]. Available: <https://www.typescriptlang.org/>. [Accessed 12 05 2020].
- [۳] I. Zviagin, "What is Framework in Software Engineering?," 20 December 2019.[Online].Available: <https://gbksoft.com/>
- [۴] Architecture - Angular, [Online]. Available: <https://angular.io/> ". [Accessed 12 05 2020].
- [۵] AngularAir, "Angular Air Episode 0: The Angular Team," 14 November 2014. [Online]. Available: <https://www.youtube.com/watch?v=LG9VkCDbte0>. [Accessed 12 05 2020].
- [۶] K. Mysliwiec, "Documentation - NestJS," [Online]. Available: <https://docs.nestjs.com/>. [Accessed 12 05 2020].
- [۷] Introduction - Material Design, Google, [Online]. Available: <https://material.io/>". [Accessed 12 05 2020].
- [۸] Component styles - Angular, [Online]. Available: <https://angular.io/> ". [Accessed 12 05 2020].
- [۹] A. Ghahrai, "Software Development Methodologies: DevQA," 3 September 2016. [Online]. Available: <https://devqa.io/software-development-methodologies/>. [Accessed 6 May 2020].
- [۱۰] ب.زمانی، ا.فاطمی، مهندسی نرم افزار شی گرا یک متدولوژی چابک یکنواخت، انتشارات دانشگاه اصفهان.
- [۱۱] L. Tan, "Unit Tests, UI Tests, Integration Tests & End-To-End Tests", 19 September 2019. [Online]. Available: <https://medium.com/@lawrey/unit-tests-ui-tests-integration-tests-end-to-end-tests-c0d98e0218a6>. [Accessed 18 June 2020]
- [۱۲] Dhormale, "Automate End to end (e2e) testing for Angular 7 using Protractor & Jasmine", 1 December 2018. [Online]. Available: <https://medium.com/@dhormale/automate-end-to-end-e2e-testing-for-angular-7-using-protractor-jasmine-4ce171aaeedc>. [Accessed 18 June 2020]

## پیوست یک (کتابخانه‌های استفاده شده در فرانت‌اند)

### وابستگی‌های اصلی

اسم پکیج	نسخه
@angular/core	۹.۱.۷
@angular/material	۹.۲.۳
@ngrx/store	۹.۱.۲
@ngrx/effects	۹.۱.۲
@ngrx/router-store	۹.۱.۲
rxjs	۶.۵.۵
moment	۲.۲۴.۰
@ckeditor/ckeditor5-angular	۱.۲.۳
@auth0/angular-jwt	۴.۰.۰
ngx-socket-io	۳.۰.۱

### وابستگی‌های محیط توسعه

اسم پکیج	نسخه
@angular/cli	۹.۱.۶
@angular/language-service	۹.۱.۷
typescript	۳.۸.۳
tslint	۵.۱۵.۰
jasmine-core	۳.۴.۰
karma	۴.۱.۰
protractor	۵.۴.۰

## پیوست دو (افزونه‌های استفاده شده در VS Code)

نام افزونه	توضیح
Angular Snippets (Version 9)	کدهای آماده‌ی انگولار با نوشتن اختصارهای آن
Angular Language Service	هوشمندسازی متغیرهای در قسمت template
Prettier - Code formatter	زیباسازی قرارگیری خطوط کدها
Bracket Pair Colorizer	رنگ‌بندی براکت‌ها در کد
Code Spell Checker	بررسی املاي کلمات نوشته شده
GitLens — Git supercharged	گیت‌داخلی برای نمایش تغییرات و توضیحات
TSLint	راهنمای نوشتار تایپ‌اسکریپت

پیوست سه (کتابخانه‌های استفاده‌شده در بک‌اند)  
وابستگی‌های اصلی

اسم پکیج	نسخه
@nestjs/core	۷.۰.۱۳
@nestjs/jwt	۷.۰.۰
@nestjs/mongoose	۷.۰.۱
@nestjs/passport	۷.۰.۰
@nestjs/platform-fastify	۷.۰.۱
Rxjs	۶.۵.۳
moment	۲.۲۴.۰
bcrypt	۳.۰.۶
jimp	۰.۱۰.۰
class-transformer	۰.۲.۳
class-validator	۰.۱۰.۰
mongoose	۵.۷.۴
passport	۰.۴.۰
fastify-helmet	۳.۰.۱
fastify-multer	۱.۵.۲
fastify-multipart	۱.۰.۵
fastify-rate-limit	۲.۴.۰
@nestjs/platform-socket.io	۷.۰.۱

وابستگی‌های محیط توسعه

اسم پکیج	نسخه
tslint	۵.۱۶.۰
typescript	۳.۷.۲
jest	۲۴.۷.۱
supertest	۴.۰.۲