دانشگاه شهید بهشتی

دانشکده مهندسی و علوم کامپیوتر

رشته مهندسی کامپیوتر گرایش نرم افزار

طراحی و پیاده‌سازی واسط کاربری تحت وب

برای ابزار تشخیص خودکار آسیب‌پذیری‌های برنامه‌های PHP

نگارش

ساجده محمدی

استاد راهنما

دکتر مقصود عباسپور

تیرماه ۱۳۹۸

دانشگاه شهید بهشتی

دانشکده مهندسی و علوم کامپیوتر

رشته مهندسی کامپیوتر گرایش نرم‌افزار

عنوان: طراحی و پیاده‌سازی واسط کاربری تحت وب برای ابزار تشخیص خودکار آسیب‌پذیری‌های برنامه‌های PHP

نگارش: ساجده محمدی

استاد راهنما: دکتر مقصود عباسپور

تاریخ و امضاء


بسم الله الرحمن الرحيم

تقدیم نامه

خدایی که آفرید

جهان را، انسان را، عقل را، علم را، معرفت را و عشق را

و به تمام عزیزانی که عشقشان را در وجودم دمید.

حضرت ولیعصر (عج) ، پدر ، مادر و همسرم

تشکر و قدردانی

از زحمات بی دریغ جناب آقای دکتر عباسپور

چکیده

تولید رابط کاربری مناسب برای ارئه‌ی خدمات تحت وب یکی از موارد مهم در جلب توجه کاربران و ایجاد یک تجربه‌ی خوب برای آنان است.

در این پروژه سعی شده است یک رابط کاربری مناسب برای ارئه‌ی خدمات ابزار تشخیص خودکار آسیب‌پذیری‌های برنامه‌های PHP طراحی و سپس پیاده‌سازی شود. برای اینکار از فریمورک Vue.jsو Bootstrap و از استاندارد REST برای اتصال به سرور استفاده شده است.

واژ گان کلیدی: رابط کاربری تحت وب، طراحی رابط گرافیکی، vue.js، UI/UX، موتور تشخیص خود کار آسیب‌پذیری

فهرست مطالب

فصل اول مقدمه

فصل دوم طراحی

فصل سوم پیاده سازی

فصل چهارم آزمون

فصل پنجم جمع بندی

فهرست تصاویر

۱ فصل اول مقدمه

۱-۱ پیش‌گفتار

با توجه به گسترش روزافزون استفاده از اینترنت و فضای مجازی یکی از معضلات اصلی در این زمینه، امنیت نرم‌افزارهای تحت وب می‌باشد. مشکلات امنیتی مانند حفظ اطلاعات شخصی کاربر، حفظ داده‌های مهم مانند اطلاعات بانکی و جلوگیری از شنود شخص ثالث از جمله مواردی است که باید رعایت نکردن برخی الگوهای امنیتی به راحتی در برنامه‌های تحت وب اتفاق می‌افتد.

وجود ابزاری مانند موتور تشخیص خودکار آسیب‌پذیری‌ها[۱]ی برنامه‌های پی‌اچ‌پی[۲] به برنامه‌نویسان مبتدی و برنامه‌نویسان حرفه‌ای که روی پروژه‌های بزرگ کار می‌کنند، این امکان را می‌دهد که ضعف‌های امنیتی برنامه‌ی خود را پیش از توسعه و ارائه تحت وب متوجه باشند و از اشتباهات رایج برنامه‌نویسی جلوگیری کنند. گرچه ارائه Supervisorی این ابزار بدون یک رابط کاربری مناسب ممکن نیست.

۱-۲ تعریف مسئله

وجود ابزار تشخیص خودکار آسیب‌پذیری‌های برنامه‌های پی‌اچ‌پی به تنهایی کافی نیست. این برنامه باید در یک قالب مناسب در اختیار کاربران قرار بگیرد. بنابراین تصمیم بر طراحی یک رابط کاربری[۳] تحت وب برای این ابزار گردید.

در قدم اول باید نیازهای اصلی سایت بررسی و با نمونه‌های موجود (در صورت وجود) مقایسه می شد. در این مرحله طراحی اولیه با استفاده از این اطلاعات صورت می‌گیرد. طراحی شامل مواردی مانند تعداد صفحات، چینش صفحات و دسترسی ها، انتخاب رنگ‌ها و المانت‌ها[۴] و طراحی اولیه صفحات می‌باشد.

---

[۱] Smart AEG (automatic exploit generation)

[۲] PHP

[۳] User Interface

[۴] Elements

در قدم بعدی با توجه به بررســـی‌های صـــورت گرفته باید تکنولوژی مورد اســـتفاده تعیین شـــود. امروزه فریمورک‌های متنوعی برای پشتیبانی از برنامه‌نویسی فرانت–اند[۱] بوجود آمده است که استفاده از آن‌ها به مراتب بهتر از استفاده از اچ‌تی‌ام‌ال[۲] می‌باشد. هم‌چنین روش اتصال به سرور و انتقال اطلاعات باید توافق شود.

گام بعدی را می‌توان آزمون نرم‌افزار[۳] توسـعه‌داده دانسـت. در این مرحله باید کاربری نرم‌افزار در صـفحات مختلف تست شود. همچنین باید از یکپارچگی اطلاعات و اتصال درست به بک–اند[۴] نیز اطمینان حاصل گردد. در این مرحله باید به رفع اشکالات بپردازیم تا از صحت کامل نرم‌افزار اطمینان حاصل کنیم.

---

[۱] Front-end
[۲] html
[۳] Software Testing
[۴] Back-end

۲ فصل دوم طراحی

۲-۱ مقدمه

تجربه‌ی کاربری (UX[۱]) به طور کلی شامل موارد زیر می‌شود[1]:

- شناخت کاربر (User Research)
- معماری اطلاعات (Information Architecture)
- قابلیت استفاده (Usability)
- طراحی سرویس (Service Design)

حال به شرح هرکدام از این موارد می‌پردازیم.

الف) **شناخت کاربر**: از طریق User Research در مورد کاربران، رفتارهایشان، اهداف آن‌ها، انگیزه‌ها و نیازهایشان یاد می‌گیریم؛ اینکه مردم چگونه به سیستم نگاه می‌کنند، رویکردشان در مورد مشکلات چیست و احساسشان در تعامل با سیستم چگونه است. در نظر داشته باشد که اگر طراحی یک سیستم را نسبت به تجربه‌ی شخصی و مفروضات خودمان انجام دهیم، ممکن است پرژه شکست بخورد.

با توجه به محدود بودن مخاطبین این نرم‌افزار و مشخص بودن هدف اصلی که ارائه‌ی یک خدمت خاص است، طراحی با توجه به سایت‌هایی با کاربرد مشابه و نظر استاد راهنما انجام شد و تحقیقات میدانی صورت نگرفت.

ب) **معماری اطلاعات**: اگر کاربر در مراجعه به نرم‌افزار با حجم خیلی زیادی از داده در سایت مواجه شود تجربه‌ی کاربری بدی را تجربه خواهد کرد. در همین راستا در این قسمت به ساختاربندی، برچسب-گذاری و سازمان‌دهی محتوا می‌پردازیم که با هدف دسترسی سریع و دقیق به مواردی که کاربر به آن‌ها نیاز دارد، انجام می‌شود.

ج) **قابلیت استفاده**: در این قسمت باید به این موضوع توجه کنیم که کاربری چه انتظاراتی از سایت دارد و باید چه امکاناتی را در اختیار کاربر قرار دهیم تا با نیازهایش مطابقت کند.

---

[۱] User Experience

د) **طراحی سرویس**: در این بخش یک ورژن/ پیش‌نویس از طراحیست که به طراح اجازه می‌دهد سریع و بدون اتلاف هزینه ایده‌های خود را پیاده‌سازی کند و تغییرات مورد نیاز را انجام دهد.

۲-۲طراحی اولیه

در طراحی نرم‌افزار ابتدا باید به بررسی نمونه‌های موجود پرداخت. گرچه با در نظر گرفتن تک‌منظوره بودن سایت، مورد کاملا مشابه پیدا نشد. باید در نظر داشت وقتی یک سایت فقط با یک مورد کاربری خاص وجود دارد، عموما مراجعینی به آن سایت ختم می‌شوند که دقیقا به دنبال همان کاربری می‌باشند. این نکته در مراحل بعدی هم در نظر گرفته شده است.

با این وجود نمونه‌هایی از سایت‌های فرشگاهی یا خدماتی موفق مورد بررسی قرار گرفت و نیازهای سایت به طور کلی به دو دسته‌ی عمومی و انحصاری تقسیم شدند.

بخش عمومی شامل این موارد است:

- ثبت نام و ورود[۱] به سایت
- صفحه‌ی اصلی (خانه[۲])
- صفحه‌ی پروفایل[۳]

بخش انحصاری را می‌توان مربوط به کاربری اصلی سایت دانست:

- ثبت کد (سفارش) جدید با امکان آپلود کد
- مشاهده‌ی تمامی کد های ثبت‌شده و نتایج هرکدام با امکان دانلود کد و گزارش
- خرید پکیج ارسال کد (مربوط به نقشه‌ی درآمد[۴] سایت)

همچنین ادمین باید بتواند به صورت دستی گزارشی را آپلود یا ادیت کند و پکیج‌های موجود در سایت را مدیریت کند.

---

[۱] Login, Register
[۲] Home Page
[۳] Profile Page
[۴] Business Plan

۲-۳چینش صفحات

چینش صفحات به این شکل انجام شد:

- **صفحه‌ی ا صلی:** در این صفحه باید پیغامی مانند خوش‌آمد به سایت وجود دا شته با شد. همچنین کاربر باید به سمت صفحه‌ی ورود هدایت شود. در این صفحه بهتر است توضیحی درباره‌ی استفاده از سایت هم وجود داشته باشد.
- **صفحه‌ی کاربر:** اگر کاربر وارد شده، یوزر[۱] سایت با شد، باید پس از ورود به این صفحه هدایت شود. این صفحه شامل زیرصفحه‌های مورد نیاز کاربر است. زیرصفحه‌ی پروفایل که به کاربر دسترسی به اطلاعات و همچنین تغییر آن‌ها را بدهد. زیرصفحه‌ی ایجاد سفارش که به کاربر اجازه بدهد کد جدید ثبت کند. زیرصفحه‌ی گزارشات که به همه‌ی کدهای ثبت شده دسترسی بدهد و کاربر بتواند هرکدام را مشاهده و در صورت دریافت جواب آن را دانلود کند. زیرصفحه‌ی خرید بسته که کاربر بتواند یک بسته‌ی طلایی خریداری کند.
- **صفحه‌ی مدیر:** اگر کاربر وارد شده، ادمین[۲] با شد، باید به این صفحه هدایت شود. این صفحه شامل زیرصفحه‌های مورد نیاز مدیر است. زیرصفحه‌ی گزارش‌ها که مدیر می‌تواند گزارش‌های ثبت شده در سایت را ببیند و پا سخ آن‌ها را ادیت کند. زیر صفحه‌ی بسته‌های سایت که مدیر باید بتواند این بسته‌ها را مدیریت کند. یعنی یک بسته را حذف یا بسته‌ی جدید ایجاد کند.

۲-٤طراحی صفحات

بعد از اینکه تعداد و نوع صفحات م شخص شد می‌توانیم طراحی آن‌ها را روی کاغذ شروع کنیم. لازم به ذکر است که این طراحی ممکن است در آینده طبق سلیقه‌ی طراح یا امکانات ابزار پیاده‌سازی تغییرات زیادی داشته باشد. از آوردن تصاویر این بخش به دلیل افزایش حجم مطالب و آماده بودن رابط کاربری در بخش بعد صرفنظر شد.

۲-۵طراحی واسط کاربری

در این مرحله طراحی گرافیکی انجام می شود و شکل ظاهری نرم‌افزار با جزئیات خوبی طراحی می شود. ر این قسمت عکس‌ها و آیکون‌ها و تم[۳] سایت انتخاب می‌شود. معمولا فایل‌های خروجی این مرحله با فرمت‌های رایج عکس می‌باشد که در مراحل برنامه‌نویسی به کد تبدیل می‌شوند. لازم به ذکر است نتایج این مرحله به طراحی نهایی سایت شباهت بسیار زیادی خواهد داشت.

---

[۱] User

[۲] Admin

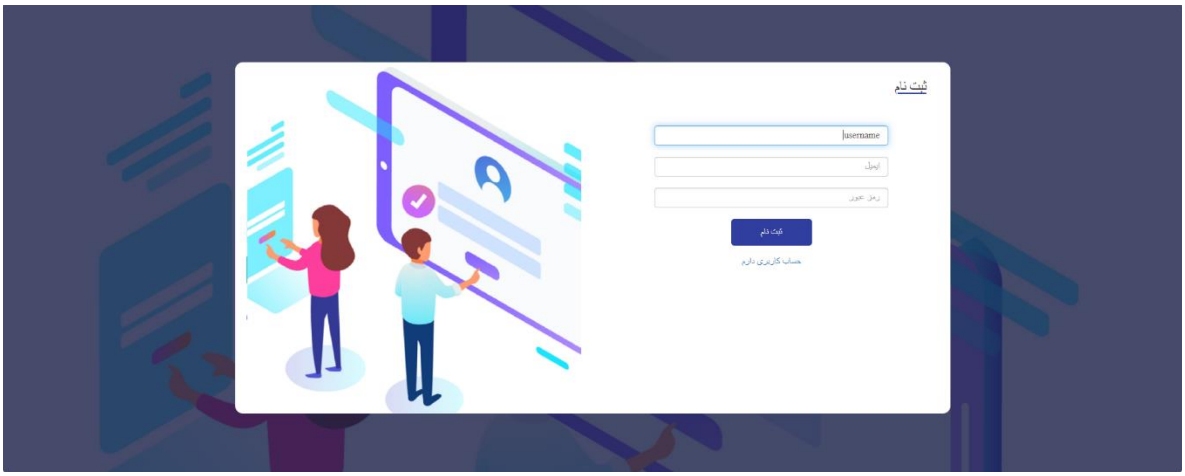
[۳] Theme

انتخاب رنگ ها، نوع چیدمان ظاهری اطلاعات، اندازه‌ی فونت، عکس های به کار رفته و ... همگی بر سرعت و راحتی استفاده از نرم‌افزار و ایجاد تجربه‌ی کاربری لذت‌بخش تاثیرگذار هستند.

شکل ۲-۱ : طراحی نهایی صفحه‌ی اول(Home Page)

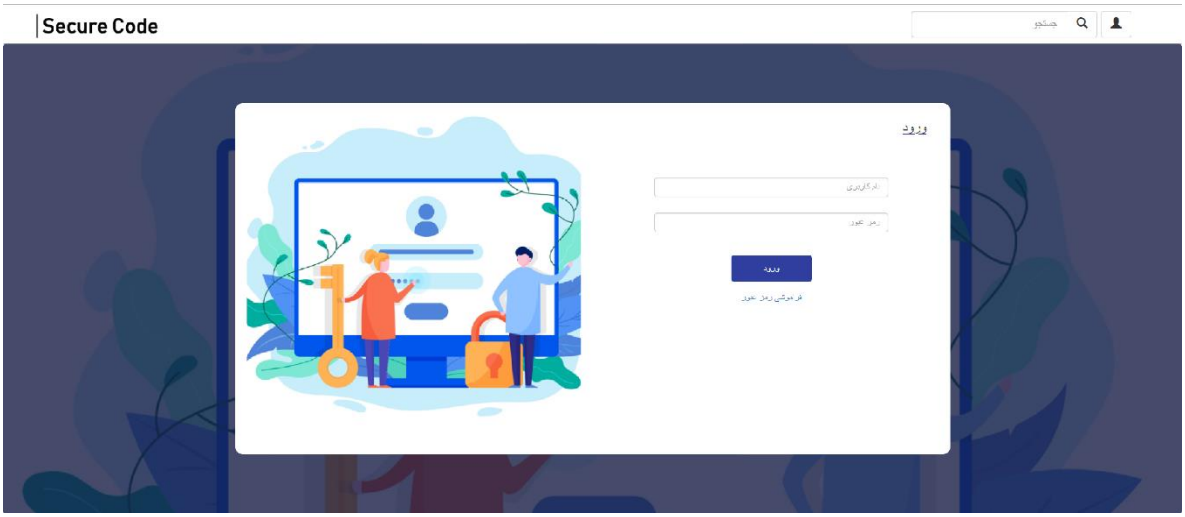
در صفحه‌ی اصلی پیغام خوش‌آمدگویی، دکمه‌هایی برای ثبت‌نام و ورود در سایت و نحوه‌ی استفاده از سایت قرار داده شده است. هدر[۱] و فوتر[۲] اصلی نیز در این صفحه طراحی شده و در همه‌ی صفحات تکرار می‌شود.

[۱] Header
[۲] Footer



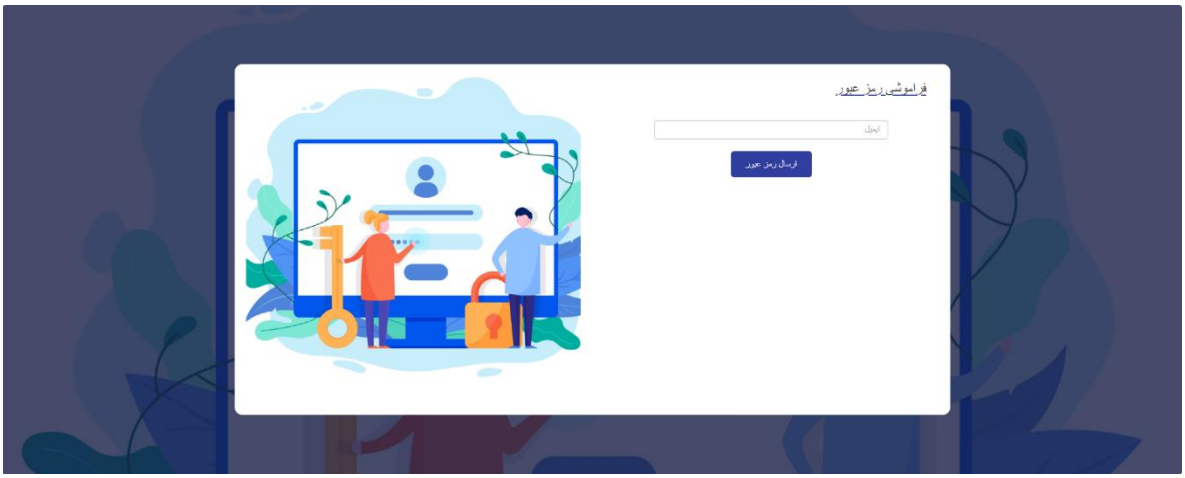
شکل ۲-۲: طراحی نهایی صفحه‌ی ثبت نام

صفحه ی ثبت‌نام، ورود و فراموشی رمز عبور مشابه یکدیگر طراحی شده اند.



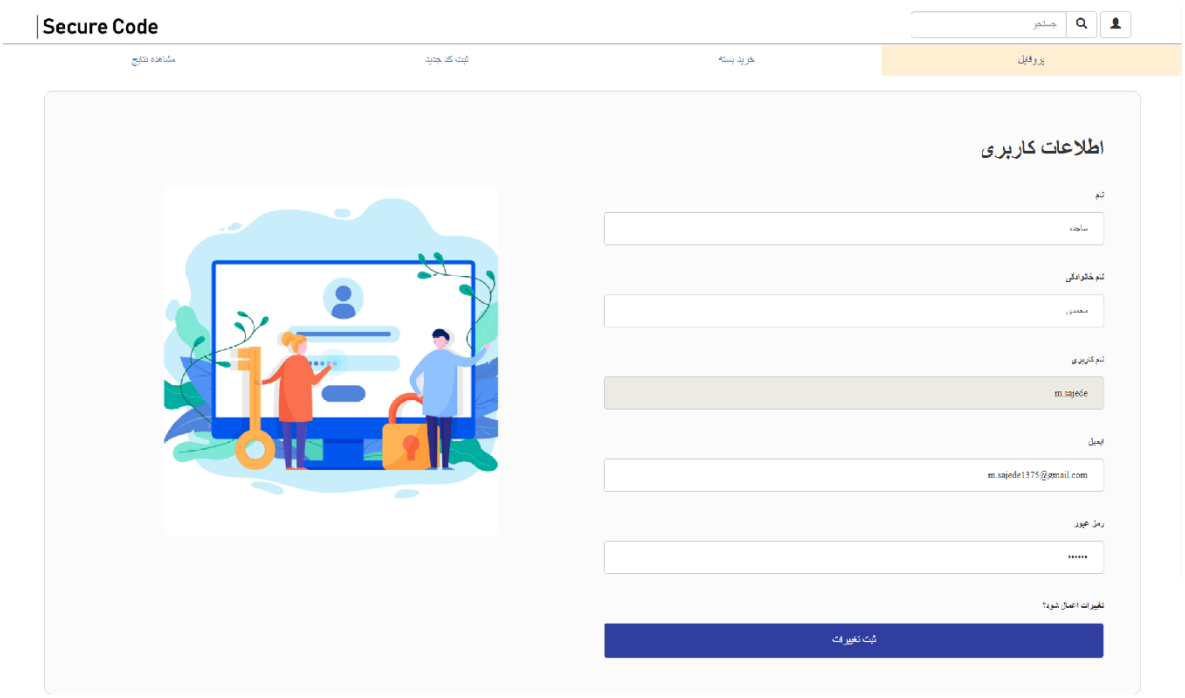
شکل ۲-۳: طراحی نهایی صفحه‌ی ورود

هدر و فوتر هر سه صفحه مشابه صفحه‌ی اصلی خواهد بود و تصویر آن‌ها در طراحی باقی صفحه‌ها تکرار نشده است.



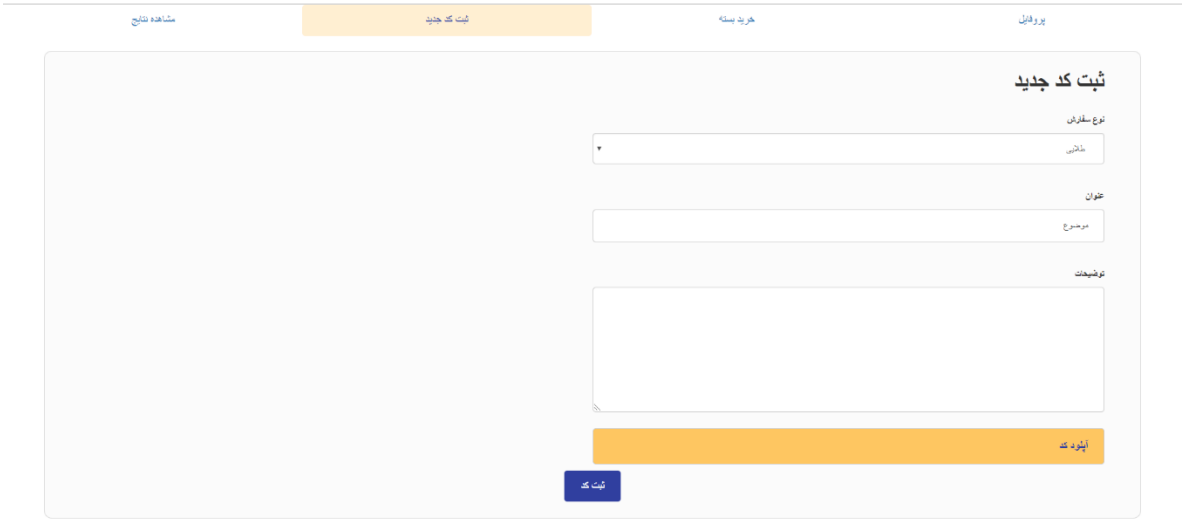
شکل ۲-۴: طراحی نهایی صفحه‌ی فراموشی رمز عبور

صفحه‌ی بعدی صفحه‌ی کاربر با تمام زیر صفحه‌های آن است.



شکل ۲-۵: طراحی نهایی صفحه‌ی پروفایل کاربر

شکل ۲-۶: طراحی نهایی صفحه ی خرید بسته ی کاربر

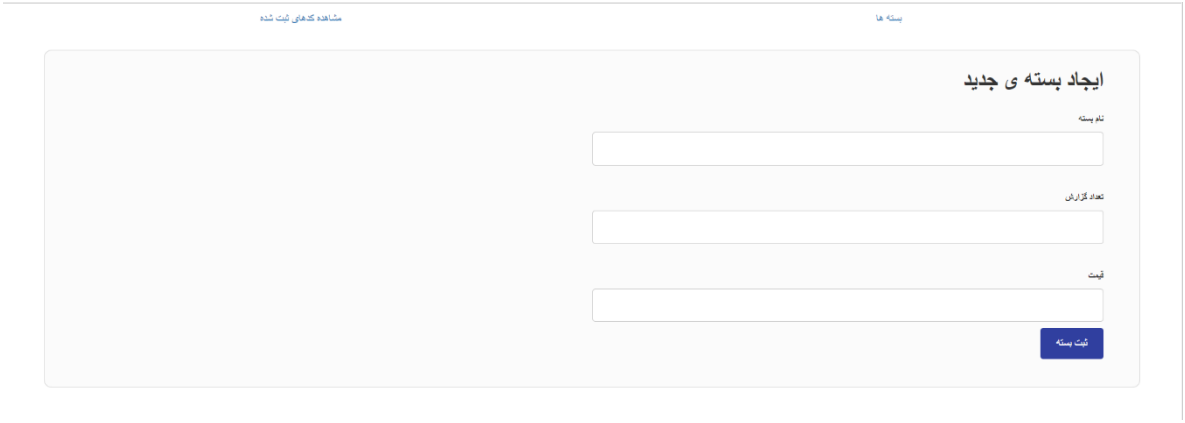


شکل ۲-۷: طراحی نهایی صفحه ی ثبت کد جدید کاربر

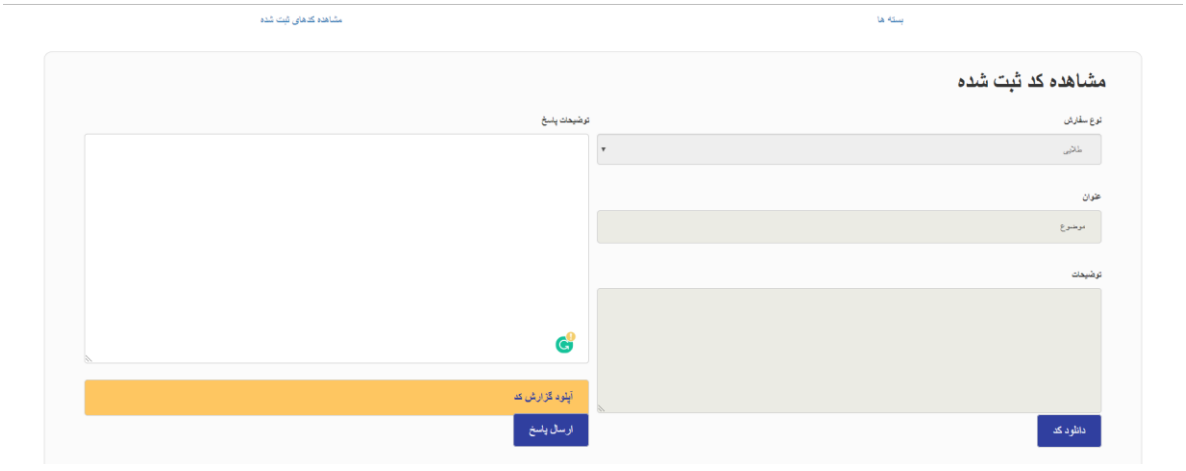
شکل ۲-۸: طراحی نهایی صفحه ی مشاهده کدهای ثبت شده ی کاربر

سپس طراحی صفحه‌های ادمین با زیرصفحه‌های آن صورت گرفت.

شکل ۲-۹ : طراحی نهایی صفحه ی مدیریت بسته های مدیر



شکل ۲-۱۰: طراحی نهایی صفحه ی ایجاد بسته ی جدید مدیر



شکل ۲-۱۱: طراحی نهایی صفحه ی ارسال پاسخ مدیر

۳ فصل سوم پیاده‌سازی

۳-۱ مقدمه

در این مرحله باید پیاده‌سازی رابط کاربری آغاز شود. در ابتدا باید تصمیم بگیریم که از چه فریمورک[۱]ی استفاده کنیم. با توجه به امکاناتی که فریمورک انتخاب شده در اختیار ما قرار می‌دهد در مراحل بعدی روش ارتباط با سرور، معماری اطلاعات سمت کاربر و پیاده سازی انجام می شود بهتر است قبل از شروع تعدادی از اصطلاحات و امکانات معرفی شود.

أ) **فریمورک برنامه نویسی**: هر فریمورک را می‌توان مجموعه‌ای از کدهای از پیش آماده، کتابخانه‌های برنامه نویسی و قوانین توسعه‌ی نرم‌افزار دانست که به برنامه‌نویس کمک می‌کند بسیاری از کدهای تکراری که در اکثر برنامه‌ها وجود دارد و مورد نیاز می باشد را بازنویسی نکند. همچنین معمولا نوشتن کدها در یک فریمورک امنیت را بهبود می‌بخشد، سرعت توسعه را افزایش می‌دهد و کمک می‌کند تا توسعه‌پذیری کد بهبود یابد. بنابراین هر برنامه‌نویس جدا از زبان برنامه‌نویسی که با آن فعالیت می‌کند، با انتخاب فریمورک مناسب مواجه است. تنوع فریمورک‌های موجود برای هر زبان برنامه‌نویسی متفاوت است، مثلا برای زبان php فریمورک‌هایی مثل Cake PHP, Zend, Symfony, Laraval وجود دارند.[2]

ب) **رابط برنامه‌نویسی نرم‌افزار کاربردی**: Aplication programing Interface (API) یا به طور خلاصه رابط برنامه‌نویسی، رابط بین یک کتابخانه یا سیستم‌عامل و برنامه‌هایی است که از آن تقاضای سرویس می‌کنند. رابط کارکردهایی را تعریف می‌کند که کتابخانه یا سیستم‌عامل می‌تواند ارائه دهد و مفهومی مجرد است.این کارکردها عموما در قالب یک کتابخانه پیاده‌سازی می‌شوند.

به طور کلی به مجموعه‌ای از توابع و رویه‌ها که به برنامه‌ی کاربردی اجازه‌ی دسترسی و استفاده از ویژگی‌ها یا داده‌های یک نرم‌افزار را می‌دهد، API آن نرم‌افزار گفته می شود. نرم‌افزار ارائه‌دهنده‌ی API می‌تواند یک سایت اینترنتی، یک سیستم‌عامل یا هر سرویس دیگری باشد.[3]

ج) **نشانه گذاری شی جاوااسکریپت**: JavaScript Object Notation (JSON)، یک استاندارد باز متنی سبک برای انتقال داده‌ها است به گونه‌ای که برای انسان نیز خوانا باشد. JSON از زبان اسکریپت‌نویسی جاوااسکریپت در نشان دادن ساختمان‌داده‌های ساده و آرایه‌های انجمنی مشتق شده است. با وجود ارتباط

---

[۱] Framework

عمیقی که با جاوا ا سکریپت دارد، JSON م ستقل از زبان ا ست و مف سرهایش تقریبا برای هر زبانی موجود است.[4]

د) **ز بان نشـــا نه گذاری گســـترش پذیر**: eXtensible Markup Language (XML) نوعی ز یان نشانه‌گذاری گسترش‌پذیر ا ست که قالب کلی نشانه‌گذاری متن‌های رایانه‌ای را تعیین می‌کند. به طوری که این زبان، هم برای انســـان هم برای ماشـــین خوانا باشـــد. در حال حاظر گوناگونی، میزان و ابعاد فراوان به کارگیری XML در اغلب زمینه‌ها و ساختارهای اینترنت امروزی چشم‌گیر است.

۳-۲انتخاب فریمورک

برای انجام این پروژه فریمورک Vue.js انتخاب شده ا ست. این فریمورک تو سط اوان یو[۱] تو سعه داده شد که قسمت‌هایی از فریمورک Angular.js را که بیشتر دوست داشت استخراج و فریمورکی بسیار سبک‌تر ایجاد کرد. به همین دلیل Vue.js را شامل بهترین خصوصیات Angular.js و React می‌دانند.[5]

از مزیت‌های اصلی این فریمورک می‌توان به موارد زیر اشاره کرد: [6]

- **حجم کم و سرعت بالا**: این فریمورک بسـیار سـبک اسـت. حجم آن در حال حاظر چیزی حدود ۱۸ تا ۲۱ کیلوبایت ا ست که بلافا صله پس از ن صب قابل ا ستفاده ا ست. همچنین کدهای نو شته شده در این فریمورک حجم بسیار کمتری نسبت به فریمورک‌های دیگر دارد که در نهایت منجر به سریع‌تر بودن آن است.
- **انعطاف پذیری**: این فریمورک بسیار انعطاف‌پذیر است. به این معنی که Vue.js توسعه‌دهنده‌ی وب را مجبور نمی‌کند که از روش خاصی استفاده کند یا مطابق الگوریتم خاصی عمل کند. بلکه همه یا قسمتی از امکانات آن را می‌توان متناسب با نیاز استفاده کرد. به همین دلیل هم ادغام با سایر فریمورک‌های مبتنی بر جاوا اسکریپت را تسـهیل می‌کند. یعنی کدهای این فریمورک را می‌توان یا سـایر اپلیکیشـن‌ها ادغام کرد و برای توسـعه و تغییر اپلیکیشن‌های موجود گزینه‌ی خوبی است.
- **ارتباط دو طرفه**: در مدل MVN[۲] با تغییر model باید view را نیز تغییر دهیم اما با استفاده از امکان اتصال دو طرفه‌ی طراحی شـــده در Vue.js با تغییر model به صـــورت خودکار تغییرات در view اعمال می‌شـــود و برعکس، این امکان موجب صرفه‌جویی در زمان و سادگی کار می‌شود.

---

[۱] Evan You

[۲] Model-View-Controller

- **سادگی**: یادگیری این فریمورک نسبت به سایر فریمورک‌ها ساده‌تر است و این یکی از دلایل محبوبیت آن به شمار می‌رود. Vue.js مستندات بسیار کاملی دارد و کسی که با اچ‌تی‌ام‌ال و جاوا اسکریپت آشنایی داشته باشد، به راحتی می‌تواند شروع به استفاده از این فریمورک کند.

علاوه بر Vue.js در انجام پروژه از فریمورک Bootstrap استفاده شده است. مزیت اصلی این فریمورک مطابقت آن با تمامی مرورگرهای استاندارد و طراحی واکنشگرا[1]ی آن است.

۳-۳انتخاب روش اتصال به سرور

روش معمول برای ارتباط با دو نرم‌افزار تحت شبکه با یکدیگر (مانند رابط کاربری و سرور) ارسال فایل‌هایی با فرمت XML یا JSON است. به دلیل فراگیرتر شدن و به روزتر بودن مفهموم REST فرمت JSON انتخاب شد. لازم به توضیح است که REST مجموعه‌ای از استانداردها و روش‌های ارسال داده‌ها عموما به فرمت JSON تحت شبکه به منظور سازمان‌دهی تعاملات مابین سیستم‌ها است. در واقع RESR یک پروتکل نیست بلکه اصولی برای استفاده از پروتکل HTTP می‌باشد.[7]

فریمورک Vue.js کتابخانه‌ای به نام vue-resource دارد که از پروتکل HTTP و فایل JSON پشتیبانی می‌کند. با استفاده از توابع این کتابخانه، می‌توان به سادگی اشیا در Vue.js را تبدیل به فرمت JSON کرد و اطلاعات دریافت شده را به راحتی به شی Vue.js تبدیل نمود. این کتابخانه امکانات دیگری مانند تغییر در header، تنظیم parameter و سایر امکاناتی که برای ارتباط با یک نرم‌افزار RESTful نیاز است را نیز فراهم می‌کند.[8]

```
import VueResource from 'vue-resource';

Vue.use(VueResource);
Vue.http.options.root = 'http://localhost:8080/contacts/rest/';
```

شکل ۳-۱: اضافه کردن vue-resource به پروژه

[1] Responsive

۳-٤معماری اطلاعات

از آن جایی که در دسترسی به خیلی از صفحات نیاز است که کاربر به مجموعه‌ای از اطلاعات دسترسی پیدا کند، برای ذخیره و مدیریت این اطلاعات در سمت کاربر باید معماری مناسبی انجام شود که مدیریت و تغییر آن‌ها را تسهیل کند.

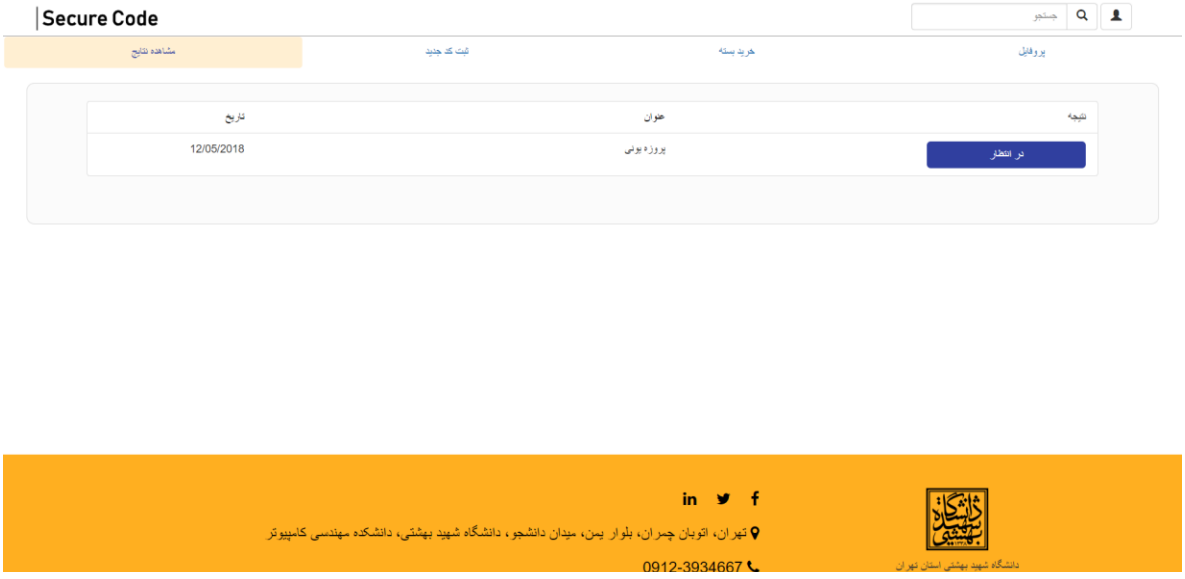
فریمورک Vue.js در این را ستا کتابخانه‌ای به نام vuex دارد که بو سیله‌ی آن می‌توان مجموعه‌ای از داده ها را به صورت متمرکز ذخیره کرد و در صورت نیاز به هرکدام دسترسی داشت. با توجه به این امکان کافی است که مجموعه‌ی این داده‌ها از سرور دریافت شود یا آن‌ها را به صورت دستی برای تست پر کنیم.[9]

```
import Vue from 'vue';
import Vuex from 'vuex';

Vue.use(Vuex);

export const store = new Vuex.Store({});
```

شکل ۳-۲: اضافه کردن vuex به پروژه

۳-۵ پیاده سازی

در این قسمت پیاده‌سازی یکی از صفحات را که شامل استفاده از همه‌ی قسمت‌های قبل می‌باشد، بررسی می‌کنیم. صفحه‌ی مورد نظر صفحه مشاهده نتایج کاربر است.



شکل ۳-۳: صفحه ی مشاهده نتایج کاربر

همانطور که قبلا ا شاره شد، فوتر و هدر یکبار طراحی شده و در همه‌ی صفحات تکرار می شود. همچنین قسمت هدر کاربر نیز برای همه‌ی زیرصفحه‌ها یکسان است. برای مدیریت قسمت‌های ثابت بر اساس آدرس صفحه[1] از یک کتابخانه‌ی دیگر به نام vue-router اســتفاده میکنیم. شــکل کامل main.js پس از اضــافه کردن همه‌ی کتابخانه‌ها به صورت زیر می‌باشد.

```
import Vue from 'vue';
import App from './App.vue';
import VueResource from 'vue-resource';

import VueRouter from 'vue-router';
import { routes } from './routes';

import { store } from './store/store';

Vue.use(VueResource);
Vue.http.options.root = 'http://localhost:8080/contacts/rest/';

Vue.use(VueRouter);
const router = new VueRouter({
  mode: 'history',
  routes
});

new Vue({
  el: '#app',
  router,
  store,
  render: h => h(App)
})
```

شکل ۳-۴: فایل main.js با افزودن کتابخانه های مورد نیاز

سپس باید در فایل routes.js مسیرهای مورد نظر برای سایت و زیرصفحه‌ی هر مسیر مشخص شود.

```
{ path: '/user', component: User, children:[
    { path: '', component: UserProfile},
    { path: 'new', component: UserNewCase},
    { path: 'list/:page', component: UserListCase},
    { path: 'case/:id', component: UserSeeCase},
    { path: 'packs', component: UserPackags},
  ]},
```

شکل ۳-۵: تعریف مسیرها در routes.js

هرکدام از UserSeeCase و.... یک کامپوننت Vue می‌باشد که در ادامه به آن می‌پردازیم.

لازم به ذکر ا ست که بنابه ساختار webpack-simple که یکی از تمپلیت‌های Vue.js می‌با شد که ما در اینجا اسـتفاده کردیم، فقط یک صـفحه‌ی index.html وجود دارد و کامپوننت اصـلی که App.vue نام دارد در آن

[1] url

لود[1] می‌شود. بنابراین فوتر و هدر اصلی سایت را می‌توان در App.vue تعریف کرد و برای دسترسی به بقی‌ی کامپونت‌ها از تگ router-view استفاده میکنیم که کامپونت نظیر شده در routes.js را لود می‌کند.

```
<template>
  <div id="app" class="container-fluid">
    <app-header></app-header>
    <router-view id="viewContainer"></router-view>
    <app-footer></app-footer>
  </div>
</template>

<script>
import Header from "./components/Header";
import Footer from "./components/Footer";

export default {
  name: 'app',
  components:{
    appHeader: Header,
    appFooter: Footer
  },
  data () {
  },
  mounted() {

  }
}
</script>

<style>
</style>
```

شکل ۳-۶: نگاه کلی به کامپونت App.vue

همانطور که دیده می‌شود، هر کامپونت شامل سه قسمت اصلی می‌باشد:

- template: در این بخش کدها به زبان html با استفاده از تگ‌های html یا تگ‌های تعریف شده توسط vue می‌باشد. همچنین vue شامل attributeهای تعریف‌شده‌ای مانند v-if و v-for می‌باشد که با تمامی تگ ها قابل استفاده است.
- script: زبان این بخش را می‌توان معادل جاوا سکریپت دانست گرچه در عمل از آن فراتر است. در این قسمت data، methods، components و توابع خاصی مانند mounted قابل تعریف و استفاده می‌باشد.

[1] load

- style: این قسمت شامل کدهای css می‌باشد که اگر از کلمه‌ی scoped استفاده شده باشد تنها مربوط به همین کامپوننت و در غیر اینصورت مربوط به کل کامپوننت‌ها می‌باشد.

یکی از توابع خاص قسمت script تابع mounted می‌باشد که مربوط به life cycle کامپوننت‌های UI و لحظه‌ی load شدن یک کامپونت می‌باشد. این تابع مناسب‌ترین قسمت برای ارتباط با بک‌اند برای دریافت اطلاعات می‌باشد.

```
mounted() {

  this.$http.get('/session',
    {
      headers: {'SessionID': this.$store.getters.sessionId}
    }).then(
    response => {
      // success callback
      this.$store.commit('sessionId', response.headers.get('SessionID') );
    },
    error => {
      // error callback

    });

}
```

شکل ۳-۷: تابع mounted در App.js

در این تابع یک درخواست به سرور با uri اضافه شده‌ی session/ ارسال می شود. این درخواست برای تنظیم sessionId بین کاربر و سرور می‌باشد تا یکپارچگی اطلاعات در صفحات مختلف حفظ شود. این id در ابتدا دریافت و سپس در sessionId که یکی از ماژول‌های vuex می‌باشد ذخیره می شود تا در صورت نیاز به ارتباط با سرور در سایر کامپوننت‌ها به عنوان Id منحصر به فرد یک نشست در header تنظیم و ارسال شود.

تمامی کامپوننت‌ها به همین ترتیب و با استفاده از کتابخانه‌های معرفی شده تنظیم شده و عمل می‌کنند.

٤ فصل چهارم آزمون نرم‌افزار

٤-١ مقدمه

با توجه به ا ستفاده از فریمورک bootstrap می‌توان گفت یه یکی از اهداف ا صلی واکن شگرا بودن سایت ا ست که برای اطمینان از آن باید نرم‌افزار را روی صفحات مختلف امتحان کرد. م سئله‌ی دیگر اطمینان از صحت ارتباط با سرور و یکپارچگی داده‌ها می‌باشد.

٤-٢ کاربری در دستگاه‌های مختلف

در این بخش صفحات مختلف سایت در پلتفرم‌های مختلف ت ست و از زیبایی و خوانایی اطلاعات اطمینان حاصل شد.

به طور مثال صفحه‌ی اصلی در مدل iPad به صورت زیر نمایش داده خواهد شد که مطلوب می‌باشد:

شکل ۴-۱: صفحه ی Home page در مدل ipad

٤-٣یکپارچگی داده

در زمان ارائه‌ی این پایان‌نامه API سمت سرور هنوز راه‌اندازی ن شده ا ست و امکان ت ست وجو ندا شت. اگرچه امکانات سایت با داده‌های فر ضی ارزیابی شد و هر زمان که API تکمیل شود، سایت آماده برای راه‌اندازی می‌باشد.

۵ فصل پنجم جمع بندی

۵-۱بررسی محصول نهایی

محصول نهایی یک رابط کاربری تحت وب می‌باشد که قابل اتصال به یک سرویس Restful برای ابزار تشخیص خودکار آسیب‌پذیری‌های برنامه‌های PHP می‌باشد. این رابط شامل امکانات عمومی مورد نیاز یک سایت مانند صفحه‌ی ورود و ثبت‌نام و نیازهای بررسی شده برای کاربرد خواسته شده مانند خرید بسته‌های کد و آپلود یک کد ودریافت گزارش می‌باشد.

از آن‌جایی‌که برای کنترل ورژن از GitHub استفاده شده است، می‌توانید برای دسترسی به کدها، پروژه را از لینک زیر دریافت کنید:

https://github.com/sajede/final

۵-۲بروز رسانی‌های پیشنهادی

از آنجاییکه امروزه تنها ارائه‌ی یک سرویس برای جلب توجه مشتریان کافی نیست، پیشنهاد می شود در آینده خدمات یا محصولات بیشتری در سایت ارائه شود که علاوه بر گسترش دامنه‌ی مشتریان، زمینه‌ی بهبود رتبه در گوگل و پیدا شدن سایت از طریق جستجو برای کاربران جدید را فراهم می‌کند.

در نظر داشته باشد که با توجه به فریمورک Vue.js که برای توسعه‌ی رابط کاربری استفاده شد، زمینه‌ی طراحی صفحات جدید با استفاده از امکانات فعلی پیاده‌سازی شده و تغییر در سایت تسهیل شده است.

منابع

[1]uidesiign.ir/principles-of-user-interface-design/

[2]rokaweb.ir/%D9%81%D8%B1%DB%8C%D9%85-%D9%88%D8%B1%DA%A9-%DA%86%DB%8C%D8%B3%D8%AA%D8%9F/

[3]rokaweb.ir/api-%DA%86%DB%8C%D8%B3%D8%AA/

[4]rokaweb.ir/json-%DA%86%DB%8C%D8%B3%D8%AA/

[5]en.wikipedia.org/wiki/Vue.js

[6]www.lydaweb.com/category/article/vuejs/82/vue-js-%DA%86%DB%8C%D8%B3%D8%AA

[7]sokanacademy.com/courses/coding/1640/%D8%A2%D8%B4%D9%86%D8%A7%DB%8C%DB%8C-%D8%A8%D8%A7-%D9%85%D9%81%D9%87%D9%88%D9%85-restful-api

[8]www.npmjs.com/package/vue-resource

[9]www.npmjs.com/package/vuex

Abstract

Creating a user-friendly interface for providing web services is one of the key issues in attracting users' attention and creating a good user experience for them.

The project attempts to design and then implement a user-friendly interface for providing services for the smart automatic exploit generation (Smart AEG) in PHP applications. It uses the Vue.js and Bootstrap framework and the REST standard to connect to the server.

Keywords: front-end development, UI/UX, Vue.js, Smart Automatic Exploit Generation

Shahid Beheshti University

Faculty of Computer Science and Engineering

Software Engineering

Title

design and implement a user-friendly interface for providing services for the smart automatic exploit generation

By

Sajede Mohammadi

Supervisor

Dr. Maghsood Abbas Poor

June 2019