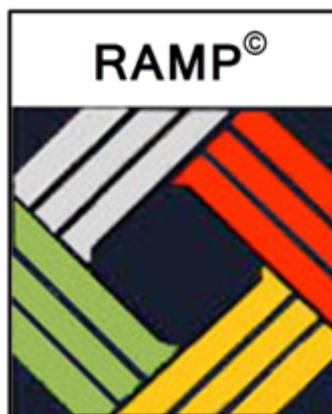


RAMP II



Projektdeltagare

Emre Demirel, Mohammed Hosin, Johan Hultin & Jonathan Lindqvist

Handledare

Reine Bergström & Anders Lindström

Introduktion	3
Specifikation	4
2.0 Restful API	4
Användargränssnittet	5
2.1 Bedömning	5
2.1.1 Riskområde 1-2	5
2.1.2 Riskområde 3-4	5
2.1.3 Riskområde 5-7	6
2.1.4 Funktionalitet	7
2.2 Resultat	8
2.3 Handlingsmodell	8
2.4 Åtgärdsförslag	8
2.5 Handlingsplan	8
2.5.1 Funktionalitet	9
Systemarkitektur	11
3.1 Översikt av arkitekturen	11
3.2 Webbapplikation	11
3.2.1 Programmering	11
3.2.2 Arkitektur	12
3.2.3 Testning	12
3.3 RESTful API	12
3.3.1 Programmering	13
3.3.2 Testing	13
3.3.3 Arkitektur	13
3.3.4 Databasen	14
3.3.5 MySQL & Hibernate	14
3.3.6 Autentisering	14
Källförteckning	16
Installation	18

Introduktion

Riskhanteringsverktyget Ramp innehåller fyra moduler där den första är Ramp I, en checklista över ergonomiska riskfaktorer för belastning; Ramp II som är en djupgående analys; en resultatmodul där resultatet kan presenteras på olika detaljnivåer och omfattning. Den sista modulen är åtgärds modulen som följer åtgärdsrekommendationer och mallar för att utforma åtgärdsplaner för riskminskning [1]. Där genomförandet av Ramp utfördes tidigare genom programmet Excel, genom att ladda ner en fil och utföra bedömningen på sin lokala dator. Detta skapade ett hinder för den som delade datat med andra, men också svårigheter att hitta rätt. Genom att göra denna analys via en webbplats bidrar det till en enklare åtkomst, också möjligheten att dela data med andra på ett mer effektivare sätt. RAMP II är en webbapplikation som kommunicerar med ett REST-API mellan en relationsdatabas. Detta möjliggör för användaren att på ett smidigt sätt skapa en bedömning eller fortsätta på en tidigare oavsett vilken enhet som används.

Specifikation

2.0 Restful API

De API metoderna som används är GET, POST och PUT där GET hämtar data från databasen. POST säkerställer att den tar emot indata för att lagra informationen på databasen, och slutligen säkerställer PUT att den uppdaterar eller ersätter det gamla värdet av befintlig data i databasen.

Användaren hade tillgång till dessa GET alternativ

- GET kunna komma åt ditt konto
- GET hämta alla checklistor som är kopplad till den inloggade användare
- GET hämta en specifik checklista som är kopplad till den användare som är inloggad
- GET hämta alla handlingsplan som är kopplad till den inloggade användare
- GET hämta en specifik handlingsplan som är kopplad till den användare som är inloggad

Användaren hade tillgång dessa anrop i POST:

- POST skapa ett nytt konto
- POST skapa en ny checklista
- POST skapa en ny handlingsplan

Användaren hade tillgång dessa anrop i PUT:

- PUT uppdatera en/flera befintlig bedömning från föregående tidpunkt
- PUT uppdatera en/flera befintlig åtgärd från föregående tidpunkt
- PUT uppdatera inmatnings datat från föregående tidpunkt

Användargränssnittet

2.1 Bedömning

Bedömningsidan innehåller alla de åtta riskområden från RAMP I och RAMP II samt inmatning av data. Gemensamt för alla riskområden är att belastningsnivån är uppdelad i tre olika nivåer med motsvarande färger. Den första färgen är grön, vilket innebär att det är låg risk för stressproblem och ansträngningar. Andra färgen är gul som har risknivån som betyder att det finns förhöjd risk att arbetstagarna utvecklar ett belastningsbesvär. Den sista färgen är röd, vilket motsvarar en hög risk, där arbetsbelastningen har en ökad risk att orsaka belastningsproblem.

2.1.1 Riskområde 1-2

Första och andra riskområdet innehåller områden inom arbetsställningar och olika arbetsrörelser, där varje fråga har en knapp som ger alternativ för antalet siffror som motsvarar den poängen. Efter att ha gjort ett urval kommer frågan att generera en färg för att visa den slutliga poängen för frågan. Där alla frågor är utrustade även med ett kommentarsfält med en text på max 65 tecken.

1. Posture

Fill in the corresponding score in the white box

Score Comment

1.1 Posture of the head - forwards and to the side

Does a clear bending of the head forwards or to the side, or twisting to the side occur, as shown in the figures, or more?

Frequency	Score
4 hours or more	2
3 to < 4 hours	1
2 to < 3 hours	1
1 to < 2 hours	1
30 minutes to < 1 hour	1
5 to < 30 minutes	0.5
< 5 minutes	0

Count: 0 / 65

1.2 Posture of the head - backwards

Does bending of the head backwards occur, as shown in the figures, or more?

Frequency	Score
2 hours or more	2
1 to < 2 hours	1
30 minutes to < 1 hour	1
5 to < 30 minutes	0.5
< 5 minutes	0

Count: 0 / 65

(Figure 1)

2. Repetitive Work

Fill in the corresponding score in the white box

Score Comment

2.1 Movements of the arm (upper and lower arm)

How are the movements of the arm generally?

Frequency	Left	Right
Constant movements mainly without pause	2	2
Frequent movements with some pauses	1	1
Varied movements, movement rate and then (up to 2/min)	0	0

Count: 0 / 65

2.2 Movements of the wrist

Do similar movements of the wrist occur?

Frequency	Left	Right
More than 20 times per minute	2	2
11 - 20 times per minute	1	1
6 - 10 times per minute	1	1
Up to 5 times per minute	0	0

Count: 0 / 65

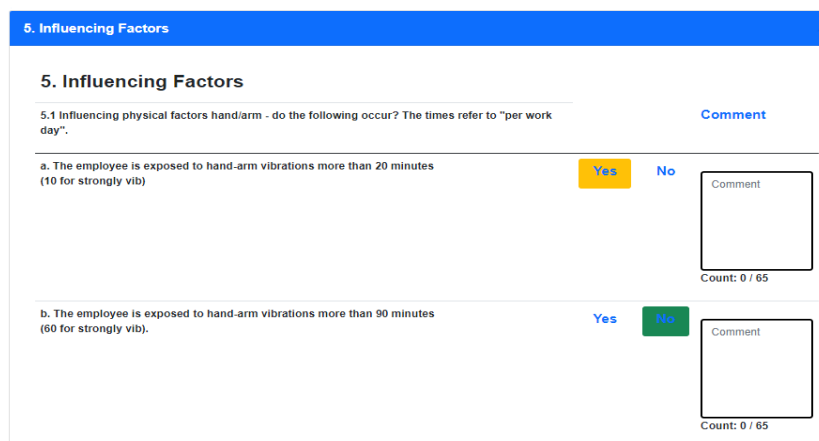
(Figure 2)

2.1.2 Riskområde 3-4

Tredje och fjärde riskområdet tar upp olika aktiva rörelser inom arbetsmiljön där användaren får besvara diverse olika frågor utifrån två tabeller där får användaren besvara de första frågorna genom att mata in värden från en tabell som visar olika värden på frekvenser inom lyftning och dragning. De resterande frågorna är Ja/Nej alternativ som är utifrån olika påståenden som sedan sammanställs och ett medelvärde och ett värsta fall beräknas. Utifrån den poängen kan man analysera det positiva eller negativa beroende av färgen som visas.

2.1.3 Riskområde 5-7

Det femte riskområdet handlar om påverkande faktorer och är uppdelat i tre delar. Gemensamt för alla sektioner är att det finns Ja/Nej alternativ, där nej motsvarar noll poäng och om ja väljs ger det användaren två följdfrågor som handlar om de fysiska arbetsbelastningar. Där den första handlar om handrörelser och den andra delen om påverkande faktorer inom helkroppsvibrationer, synförhållandet, stillastående, där användaren besvarar med Ja/Nej. Den sista delen handlar om att påverka arbetsorganisatoriska och psykosociala faktorer för diverse arbetsuppgifter. Gemensamt för alla delfrågor är att varje fråga har ett kommentarsfält med ett maximalt intervall på 65 tecken.



5. Influencing Factors

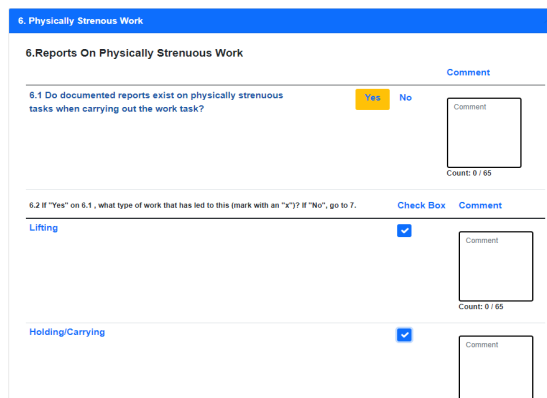
5.1 Influencing physical factors hand/arm - do the following occur? The times refer to "per work day". [Comment](#)

a. The employee is exposed to hand-arm vibrations more than 20 minutes (10 for strongly vib) **Yes** No
Count: 0 / 65

b. The employee is exposed to hand-arm vibrations more than 90 minutes (60 for strongly vib). **Yes** **No**
Count: 0 / 65

(Figur 3)

De två sista riskområdena sex och sju har en fråga och ges två alternativ antingen Ja/Nej där Ja motsvarar risknivån gul som har två poäng och nej som är lågrisk med färgen grön motsvarar noll poäng. Den första handlar om fysiskt ansträngande arbete och den andra om fysiskt obehag i leder och muskler. Varje alternativ innehåller en kommentarsfält med 65 tecken. Beroende på hur man besvarar frågorna kan det leda till totalt 10 fält, där fem av fälten innehåller namnfält på vilka personer som upplever den fysiska obehaget.



6. Physically Strenuous Work

6. Reports On Physically Strenuous Work [Comment](#)

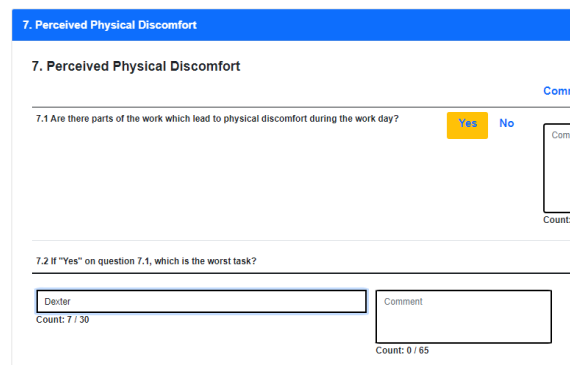
6.1 Do documented reports exist on physically strenuous tasks when carrying out the work task? **Yes** No
Count: 0 / 65

6.2 If "Yes" on 6.1, what type of work that has led to this (mark with an "X")? If "No", go to 7. [Check Box](#) [Comment](#)

Lifting ☒
Count: 0 / 65

Holding/Carrying ☒
Count: 0 / 65

(Figur 4)



7. Perceived Physical Discomfort

7. Perceived Physical Discomfort [Com](#)

7.1 Are there parts of the work which lead to physical discomfort during the work day? **Yes** No
Count:

7.2 If "Yes" on question 7.1, which is the worst task?

Dexter
Count: 7 / 30

Count: 0 / 65

(Figur 5)

2.1.4 Funktionalitet

När en auktoriserad användare utför en bedömning krävs det inte att alla inmatningsfält är ifyllda för att kunna lagra datat på databasen. Sparandet kräver dock att användaren måste ange ett unikt ID namn som däremot har en validering som ger användaren ett felmeddelande. Detta sker om användaren matar in en tom sträng eller ett upptaget ID. Det finns även en möjlighet att spara som en färdig bedömning, enda villkoret som krävs då är att hela listan är ifylld. Om användaren skulle bocka in att den var färdig skulle det ske en validering över hela bedömningen som kontrollerar om allt är ifyllt. Denna koll kommer ges ut mot användaren som ett felmeddelande på de områdena som saknar den datan. Om användaren lyckas spara bedömningen kommer allt att återställas och visa ett lyckat meddelande till användaren.

Det finns även en möjlighet för en användare att hämta befintliga bedömningar och därmed fortsätta med dem. De enda listorna som kan hämtas är de som är länkade till användaren. Hämtningen sker via textinmatning via ID namnet som de sparades på. Denna händelse har även en valideringskontroll där den kontrollerar om ID namnet stämmer överens med datat i databasen. Användaren får upp ett meddelande om hämtningen har lyckats eller inte. Om en användare hämtar en bedömning som är markerad som komplett kan man enbart läsa datat. ID namnet kan även ändras under ett sparande med en befintlig lista där den enbart uppdaterar namnet med en validering på det unika namnet.

På bedömningssidan har användaren även ett val att starta en ny bedömning från en kopia av en befintlig lista. Detta sker genom en sökning på den befintliga listan som ligger i databasen och sparandet kräver då ett nytt unikt ID namn där användaren får upp ett meddelande om det lyckats. Det sista valet som användaren har i bedömningssidan är att den kan se alla listor i en tabell som är kopplat till kontot som användaren är inloggad med. Informationen som anges i tabellen är ID namnet och datumet från när listan skapades.

Saving the assessment
Enter a name for the assessment

test ✓

Assessment completed? ✓
*Make sure you have filled in EVERYTHING in the checklist

Invalid

- Input Data is Missing Values From The First Nine Fields
- 1. Posture Missing Some Values
- 2. Repetitive Work Missing Some Values
- 3. Lifting Work Missing Some Values
- 4. Pushing And Pulling Work Missing Some Values
- 5. Influencing Factors Missing Some Values
- 6. Physically Strenuous Work Missing Values
- 7. Physical Discomfort Missing Values

Close

Close Save

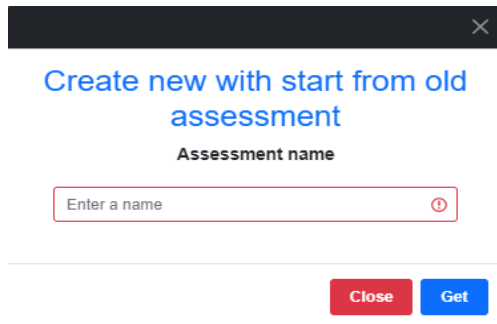
(Figur 6)

Check out all your checklists
Click Get To Access Your Checklist

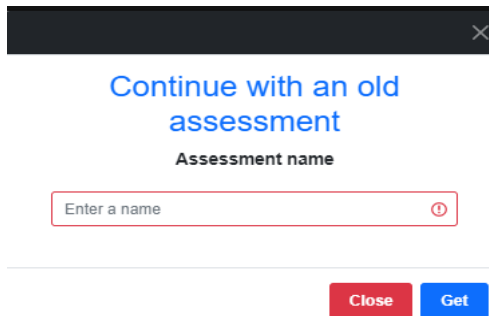
ID names	Date
Lista 1	2022-05-13

Close Get

(Figur 7)



(Figur 8)



(Figur 9)

2.2 Resultat

Resultatsidan visar bedömningen för alla frågor som användaren besvarade på bedömningssidan. Användaren kan klicka på det riskområde som de vill se bedömningen för. Det finns också en sammanfattande del som endast visar den totala riskpoängen och antalet bedömningar i riskfärgerna.

2.3 Handlingsmodell



Handlingsmodellsidan är en beskrivning av rekommenderade åtgärder för att minimera arbetsbelastningen. Innehållsmässigt har den en strukturerad information med en tabell med exempellösningar.

2.4 Åtgärdsförslag

På behandlingsåtgärd sidan finns det förslag på åtgärder som finns för att kunna minska risken för olika områden. Endast åtgärderna för de områdena som fyllts i och som fått bedömningen gul eller rött visas på sidan.

2.5 Handlingsplan

En auktoriserad användare har även tillgång till en handlingsplanssida som är kopplad till en bedömning och dess riskområden. Handlingsplanen samlar all information från bedömningen, poängen, färgen och kommentarsfältet. Varje fråga från bedömningen visas i en tabell i handlingsplanen med en åtgärdsknapp. Användaren kan lägga till en handlingsplan för varje delfråga med en åtgärd, när det ska ske, uppföljningar och vem som är ansvarig över det området

Risk factor	Assessment	Score	User comments	Planned actions	When	By whom	Ready (Date)	Follow-Up	
1. Postures									
1.1 Posture of the head - forwards and to the side		7	Behöver bättre stol	Se till att ta flera raster	2022-05-23	Jakob	2022-05-24	2022-05-30	Action Plan Details
1.2 Posture of the head - backwards		1.5							Action Plan Details

(Figur 10)

2.5.1 Funktionalitet

För att kunna använda en handlingsplan måste en auktoriserad användare hämta en befintlig bedömning. För att kunna spara en handlingsplan krävs det ett unikt ID namn. Under sparandet sker en validering som kontrollerar om det ID namnet är upptaget eller inte, i värsta fall skickas det ett felmeddelande till användaren via användargränssnittet. När en användare skapar en helt ny bedömning och går direkt till handlingsplanen kommer det ske en validering när användaren försöker spara en handlingsplan. Denna validering kontrollerar både om ID namnet är unikt och att den befintliga bedömningen är sparad. Valideringen kommer därmed visas med ett meddelande om det har lyckats eller inte.

Det finns även en möjlighet för en användare att hämta befintliga handlingsplaner och därmed fortsätta med den handlingsplan som hämtats. Dessa handlingsplaner nås enbart om man är inloggad och kan därför bara hämta de som är länkade till kontot. Hämtningen av en handlingsplan görs via ett textinmatningsfält via ett ID namn som de sparades på. Denna händelse har även en valideringskontroll som kontrollerar om det inmatade ID namnet stämmer överens med det lagrade datat. Användaren får upp ett meddelande i användargränssnittet om hämtningen har lyckats eller inte.

När användaren sparar en hämtad handlingsplan går det att ändra ID namnet som i sin tur kommer kontrolleras om att namnet är unikt eller inte. Om det nya ID namnet är unikt kommer det att visas för användaren med ett meddelande att det lyckats. Användaren kan även ersätta sin sparade bedömning genom att hämta en ny och spara det i handlingsplanen.

Det finns även ett alternativ för användaren genom att välja att starta en ny handlingsplan från en befintlig handlingsplan. Detta görs genom att den kopierar datat och sparar på ett nytt unikt ID namn. Användaren måste mata in de befintliga ID namnet för att kunna hämta upp datan som ska kopieras till de nya. Validering skes och förmedlar användaren genom att skicka ett meddelande via användargränssnittet. Användaren har även en möjlighet att se alla sina handlingsplaner som är länkade till det inloggade kontot. I användargränssnittet visas detta genom en tabell med ett ID namn och ett datum på när handlingsplanen skapades.

×

Find all Action plans

Click Get To Access Your Action Plan

ID names

Date

You have no action plans

Close

Get

(Figur 11)

×

Saving action plan

Enter a name for the action plan

Action plan name

0

Please enter a name for saving

Close

Save New

(Figur 12)

×

Create new from an old Action plan

Enter the name

Action plan name

0

Close

Get

(Figur 13)

×

Continue with an old Action plan

Enter the name

Action plan name

0

Close

Get

(Figur 14)

Action plan

Based on Ramp II assessment

Ordered by

Formed by

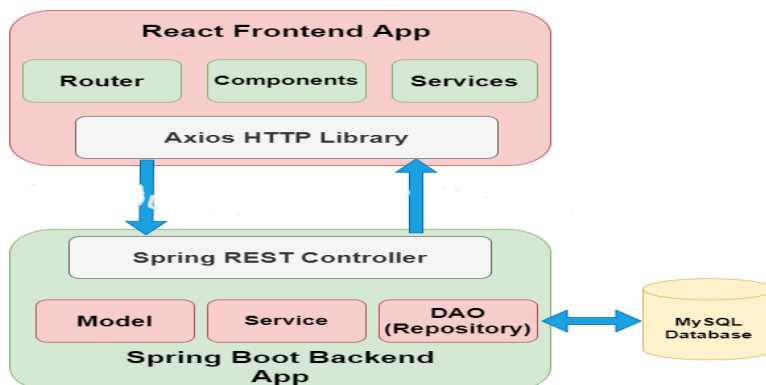
aaaa-mm-dd

Note

(Figur 15)

Systemarkitektur

3.1 Översikt av arkitekturen



(Figur 16)

Figur 16 visar en överblick av RAMP II:s arkitektur. Webbapplikationen är uppbyggd i ramverket React, som är ett JavaScript ramverk [2]. Varje sida i RAMP II motsvaras av en komponent i React. Mer detaljer om arkitekturen för webbapplikationen beskrevs i 3.22. Backend är uppbyggd i Spring boot. Spring boot är ett Java webbramverk som har en öppen källkod och är en tjänst som gör det enkelt att utveckla webbappar och andra mikrotjänster [3]. Webbapplikationen kommunicerar med backenden via RESTful API. RESTful API är en applikation programmeringsinterface som används för att skicka data mellan backenden och webbapplikationen [4]. Mer detaljer om arkitekturen för backenden beskrivs i 3.33. En MySQL databas används för att lagra information.

3.2 Webbapplikation

Ramp II webbapplikationen ger den inloggade användaren tillgång till databasens innehåll och dess funktioner. Webbplatsens huvudsyfte är att skapa ett användarvänligt gränssnitt som underlättar tillgången till att genomföra bedömningar och manipulera den befintliga datan på ett korrekt sätt.

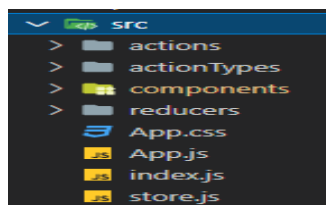
3.2.1 Programmering

Applikationens användargränssnitt är uppbyggt av ramverket React, Javascript och Redux. React är ett effektivt och flexibelt ramverk som används för att bygga användargränssnitt, där kodstrukturen är uppbyggd av en hierarki av komponenter [5]. Dataöverflödet i applikationen mellan komponenterna skedde via Redux som är det officiella React UI bindningsskiktet för att tillåta att läsa data och skicka samt uppdatera tillstånd mellan komponenterna [6]. React Hooks har även använts för att låsa fast vissa tillstånd i komponenterna, till exempel från att en modal ska synas när användaren interagerar med en knapp [6].

3.2.2 Arkitektur

När det gäller arkitekturen för webbapplikationen finns det två sektioner som delar upp programmet, det visuella gränssnittet och den logiska kodstrukturen. Interaktionen mellan användarens inmatning av data och de logiska komponenterna görs via Redux. Detta gör att dataöverföringen mellan komponenterna får en form av persistens för webbapplikationen.

Den första delen delas upp enligt de sidor som finns beskrivet i 2.1-2.5 där varje sida har React komponenter som bygger upp sidan. Komponenter som återanvänds runt om i webbapplikationen såsom Navbaren har sin egen mappstruktur. Den första delen har en mappstruktur som delats upp i flera delar baserat på de olika sidorna. Andra delen, Redux, delas upp i tre olika delar. Den första är *ActionTypes* som beskriver hur Actions ska vara definierad. De andra var Action som innehåller logiken för funktionerna. Detta utfördes med en koppling mellan *Actions* och *ActionTypes* via *Reducers* som manipulerar datat på ett korrekt sätt.



(Figur 17)

3.2.3 Testning

I frontend utfördes testningarna via try and error via alla utfall som praktiskt skulle kunna uppstå i applikationen. Alla testningar var ursprungligen tagna från användarberättelserna. Testningen delades upp i fem olika områden där den första handlade om inmatningsfälten som användaren kunde interagera med. Testningen för detta var att kontrollera att det inte uppstod några fel, alltså ogiltiga värden som inte kunde godtas. Den andra delen handlade om sparandet via användargränssnittet med den giltiga inmatnings datan på bedömningssidan, detta kontrollerades även via hämtning av data samt sparandet. Den sista var en testning för responsiviteten på applikationen, där det testades med olika upplösningar. Där Google Chrome har en funktion i inspekteringsläget där man kan välja vilken enhetsskärm som ska testas.

3.3 RESTful API

Rest API används för att spara data som skickas till databasen, även hämta ut data som man vill söka upp samt kunna uppdatera den hämtade datan som ges från databasen. API:et körs i Entity Core där Spring boot ramverket används.

3.3.1 Programmering

Backend är skriven i programmeringsspråket Java och ramverket Spring boot valdes för att bygga upp backenden. Spring Boot används för det mesta som en backend där den hanterar Rest API med databaskommunikation. Det gör att det blir bättre kvalitet på applikationer det medför att utvecklingstiden minskar. Spring boot erbjuder flera funktionaliteter som att den har en inbyggd server som kan vara Tomcat och Jetty samt att det inte kräver xml konfiguration [9][10]. Det är fristående program som gör att man inte behöver vara kopplad till en viss plattform för att skapa applikationer.

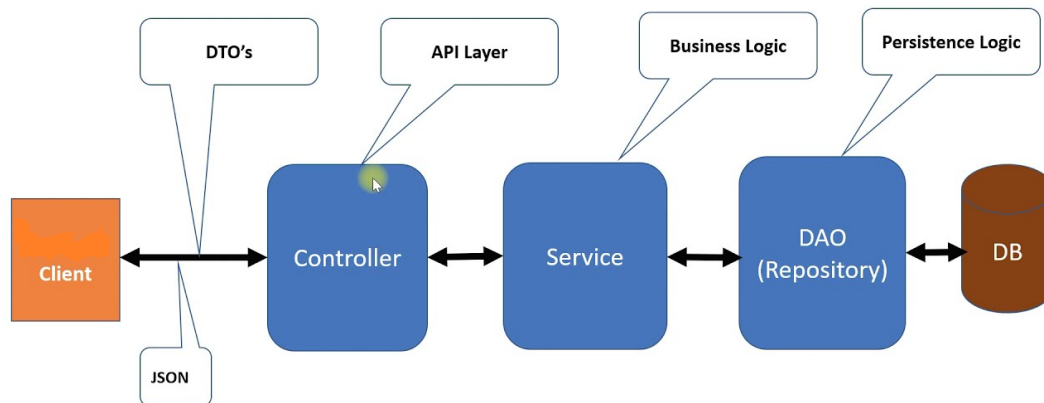
3.3.2 Testing

Testing skedde via Postman där det skrevs in en Rest API webbadress. Detta tillåter att det gick att köra olika tester med GET, POST och PUT. GET hämtar datat baserat på sökningen från Rest API:et och POST lagrar datat från anropet där man skickar en body request för att se vilken status det blir. Statusen kan vara 200 som betyder att det lyckats, 404 syftar på att det inte hittat något och 500 är när det uppstår ett fel på webbservern. PUT anropar en metod från Rest API som kan vara att den uppdaterar värdet på det lagrade datat.

3.3.3 Arkitektur

Webbapplikationen kommunicerar med backenden via RESTful API. Detta görs med hjälp av API anrop där man skickar data i JSON format till kontrollen. RAMP II projektet har tre kontroller, ActionPlanController, AuthenticationController och ChecklistController. Den första behandlar anrop relaterade till Action planen, mer information om Action planen kan hittas i 2.5. Den andra behandlar autentisering där Spring Boot Security har använts för att autentisera och verifiera att rätt och inloggad användare har tillgång till det sparade datat. Mer information om autentisering hittas i 3.36.

ChecklistController behandlar anrop relaterade till bedömningen som beskrivs mer i 2.1. Data som kontroller tar emot är DTO, *Data Transfer Object*, som används för att skicka data runt i olika områden. När kontroller har tagit emot anropet så gör den anrop till service som innehåller alla logik. Via services anrop till repository som har tillgång till persistens logik som kommunicerar med databasen. Backend är en trelagers applikation med Spring boot Security som säkerhet samt *SQL* som databas.



(Figur 18)

3.3.4 Databasen

Applikationen kräver en databas för att lagra informationen om bedömningen, handlingsplanen och en användare. Varje riskområde har sin egen tabell där varje riskområde i handlingsplanen har egna tabeller. Tabellstrukturen är baserad på bedömningen, handlingsplanen och användaren samt deras relationer. Checklisttabellen innehåller det unika id namnet samt relationer till alla tabeller för riskområdena och relation till användaren. Handlingsplantabellen innehåller unika id, vem den är gjord för, vem som har skapat det, när det skapades datum, anteckning och relationer till riskområdetablerna för handlingsplanen som är inte samma som checklistan och relation till användare.

3.3.5 MySQL & Hibernate

Databasen använder sig av MySQL [8] där anledningen till att databasen valdes var att projektet bygger på att varje behandlingsområde har en relation till kontot och behandlingsplanen. En annan fördel med MYSQL i kombination med Spring Boot är att man kan integrera med Hibernate [11], vilket ger flera fördelar där det enkelt kan kommunicera med databasen genom att skapa objekt och mappa de som entiteter. När man gör förändringar på applikationslagret sker det automatiska ändringar på databasen som uppdaterar datat.

3.3.6 Autentisering

Webbapplikationen har en autentiserings tjänst där användaren har möjligheten att skapa ett konto eller logga in för att komma åt mer funktionalitet på webbplatsen. Detta har lett till en ökad säkerhet där en inte autentiserad användare bara har tillgång till de statiska informationssidorna. När en användare är inloggad får den en token, en json web token, används för att identifiera en användare. Där varje token är unikt för varje användare [12].

En inloggad användare har möjligheten till fler sidor på webbplatsen där användaren kan skapa flera bedömningar och åtgärdsplaner som har en kommunikation mellan databasen. På backenden har Spring Boot Security blivit implementerad för att se till att enbart en inloggad användare kan kommunicera med databasen. Detta medför att varje anrop till kontrollern kontrolleras om anropet innehåller relevanta autentiseringsuppgifter för att sedan godkänna eller neka anropet [13].

Källförteckning

- [1] KTH, "Ramp - ett riskhanteringsverktyg för manuell hantering",
<https://www.kth.se/sv/mth/ergonomi/forskning/ramp-utveckling-implementering-och-spridning-av-belastningsergonomiskt-bedomningsverktyg-och-atga-1.6037>. Hämtad 2022-05-24.
- [2] React JS, "React - A JavaScript library for building user interfaces",
<https://reactjs.org>. Hämtad 2022-05-22.
- [3] Spring, "Spring Boot",
<https://spring.io/projects/spring-boot>. Hämtad 2022-05-21.
- [4] RedHat, "What is a REST API",
<https://www.redhat.com/en/topics/api/what-is-a-rest-api>. Hämtad 2022-05-22.
- [5] C-sharpcorner, "What and Why React.js",
<https://www.c-sharpcorner.com/article/what-and-why-reactjs/>. Hämtad 2022-05-22.
- [6] React Redux, "Getting Started with React Redux",
<https://react-redux.js.org/introduction/getting-started>. Hämtad 2022-05-22.
- [7] React JS, "Introducing Hooks",
<https://reactjs.org/docs/hooks-intro.html>. Hämtad 2022-05-24.
- [8] MySQL, "MySQL",
<https://www.mysql.com/>. Hämtad 2022-05-21.
- [9] Apache Tomcat, "Apache Tomcat",
<https://tomcat.apache.org/>. Hämtad 2022-06-02.
- [10] Jetty, "Jetty - Your Event Management System",
<https://jetty.se/>. Hämtad 2022-06-02.
- [11] Hibernate, "Hibernate Search",
<https://hibernate.org/search>. Hämtad 2022-05-21.
- [12] IETF, "JSON Web Token (JWT)",

<https://datatracker.ietf.org/doc/html/rfc7519>. Hämtad 2022-06-02.

[13] Spring, “Spring Security”,

<https://spring.io/projects/spring-security>. Hämtad 2022-06-02.

Appendix

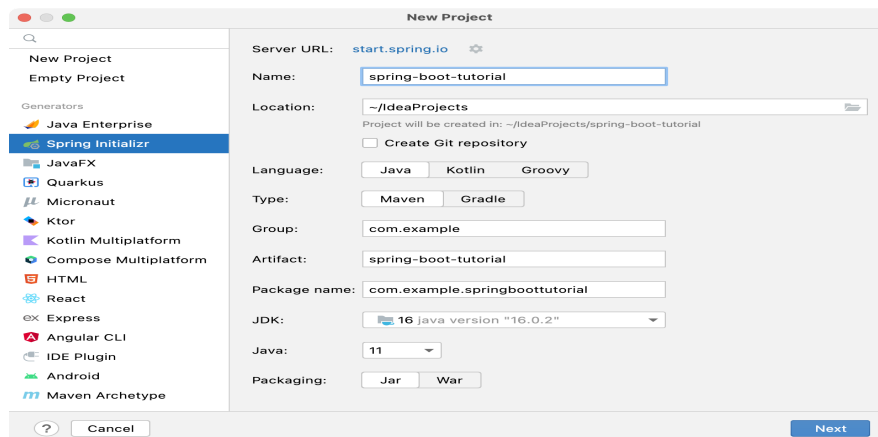
Installation

MySQL

För att ladda ner MySQL så är det via deras egen hemsida <https://dev.mysql.com/downloads/installer/>

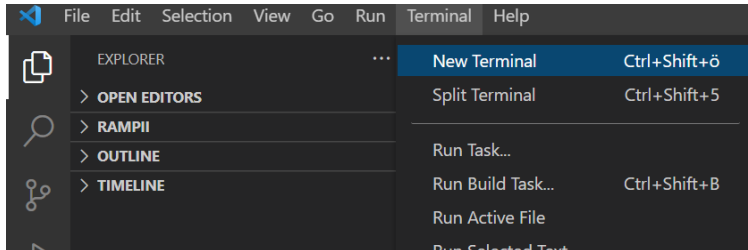
Spring Boot till IntelliJ

1. Ladda ner IntelliJ från deras hemsida <https://www.jetbrains.com/idea/download/#section=windows>
2. Från huvudmenyn välj arkiv sen gå till nytt projekt.
3. Vänstra rutan så leta efter nytt projekt så väljer du Spring Initializr.
4. Ange ett namn för ditt spring boot projektet.
5. Från JDK ladda ner den senaste version av Oracle OpenJDK.
6. Java versionen ska också väljas helst senaste versionen.

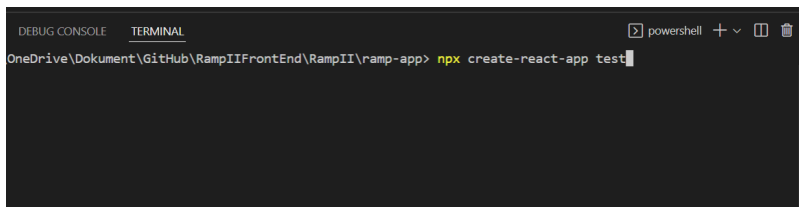


React till vsscode

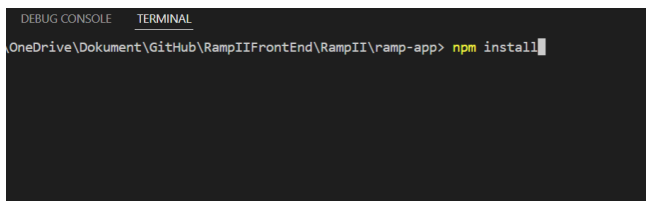
1. Ladda ner vsscode från deras egna hemsida <https://code.visualstudio.com/download>
2. Ladda ner node js från deras egna hemsida <https://nodejs.org/en/>
3. Öppna Vsscode tryck på terminalen via menyn välj på new terminal



4. Därefter välja powershell och skriv in npx create-react-app (frivillig namn)



5. Efter installationen på react appen så ska node js installeras via terminalen



Github

1. Ladda ner github desktop från deras egna hemsida <https://desktop.github.com/>
2. Öppna github desktop klicka på file sen tryck på clone repository
3. Ladda ner länken repository på backend <https://gits-15.sys.kth.se/hosin/RampIIBackEnd>
4. Ladda ner länken repository på frontend <https://gits-15.sys.kth.se/hosin/RampII>

