Vaatimusmäärittely

Ebinbodaus – Treeni ja Paino-ohjelma

Topi Korhonen

Miro Kähärä

John Ramstedt

Joonatan Lipiäinen

Sisällysluettelo

[1. Johdanto 3](#_Toc160770563)

[1.2 Tarkoitus ja kattavuus 3](#_Toc160770566)

[1.3 Tuote ja ympäristö 3](#_Toc160770567)

[1.4 Määritelmät, termit ja lyhenteet 3](#_Toc160770568)

[1.5 Viitteet 4](#_Toc160770569)

[1.6 Yleiskatsaus dokumenttiin 4](#_Toc160770570)

[2. Yleiskatsaus 5](#_Toc160770571)

[2.2 Ympäristö 5](#_Toc160770574)

[2.3 Toiminta 5](#_Toc160770575)

[2.4 Käyttäjät 5](#_Toc160770576)

[2.5 Yleiset rajoitteet 5](#_Toc160770577)

[2.6 Odotukset ja riippuvuudet 6](#_Toc160770578)

[3. Tiedot ja tietokanta 7](#_Toc160770579)

[3.2 Tietosisältö 7](#_Toc160770582)

[3.3 Käyttöintensiteetti 7](#_Toc160770583)

[3.4 Kapasiteettivaatimukset 7](#_Toc160770584)

[3.5 Tiedostot ja asetustiedostot 7](#_Toc160770585)

[4. Toiminnot 8](#_Toc160770586)

[4.2 Pääsivu ja navigointi 8](#_Toc160770589)

[4.2.2 Pääsivun toiminta step-by-step koodissa: 8](#_Toc160770590)

[4.3 Painonseuranta 9](#_Toc160770591)

[4.3.2 Painonseurannan toiminta step-by-step koodissa: 11](#_Toc160770593)

[4.4 Treenipäiväkirja 12](#_Toc160770594)

[5. Ulkoiset liittymät 14](#_Toc160770595)

[5.2 ScottPlot 14](#_Toc160770596)

[6. Muut ominaisuudet 15](#_Toc160770597)

[6.2 Suorituskyky ja vasteajat 15](#_Toc160770598)

[6.3 Saavutettavuus (availability), toipuminen, turvallisuus, suojaukset 15](#_Toc160770599)

[6.4 Ylläpidettävyys 15](#_Toc160770600)

[6.5 Siirrettävyys ja yhteensopivuus 15](#_Toc160770601)

[6.6 Operointi 16](#_Toc160770602)

[6.7 Käytettävyys (Usability), käytön tehokkuus, käyttäjien tyytyväisyys 16](#_Toc160770603)

[7. Suunnittelurajoitteet 17](#_Toc160770604)

[7.2 Standardit 17](#_Toc160770605)

[7.3 Laitteistorajoitteet 17](#_Toc160770606)

[7.4 Ohjelmistorajoitteet 18](#_Toc160770607)

[7.5 Muut rajoitteet 18](#_Toc160770608)

[8. Hylätyt ratkaisuvaihtoehdot 20](#_Toc160770609)

[9. Jatkokehitysajatuksia 21](#_Toc160770610)

[Liitteet 22](#_Toc160770611)

# Johdanto

Tämä vaatimusmäärittely on laadittu vastaamaan Ebinbodaus – Treeni ja Paino-ohjelman kehittämistarpeisiin sekä toiveisiin. Tämän dokumentin tarkoituksena on kuvata yksityiskohtaisesti sen toiminnalliset ja ei-toiminnalliset vaatimukset.

Tässä kappaleessa kuvataan yleisesti Ebinbodaus – Treeni ja Paino -ohjelman käyttötarkoitusta, käyttöympäristöä sekä keskeisiä käsitteitä.



## Tarkoitus ja kattavuus

Ebinbodaus – Treeni ja Paino-ohjelman on tarkoitettu mahdollistamaan käyttäjän painon nousua tai painon pudotusta, ohjelmisto soveltuu myös treenaajille, jotka haluavat ylläpitää kirjaa treenipainoista, viikkotreenistä ja kehityksestään, ilman kuukausi-/ jäsenmaksuja.

Käyttäjä kirjaa haluamilleen päiville sen hetkisen painonsa ja treeninsä ja edistystä seurataan visuaalisesti viivadiagrammin avulla.

## Tuote ja ympäristö

Ohjelmisto on tietokoneella käytettävä sovellus, jota käyttäjä itse päivittää ja seuraa säännöllisesti suorituksen jälkeen.

## Määritelmät, termit ja lyhenteet

Tässä on listaus määritelmistä, termeistä sekä lyhenteistä, jotka ovat oleellisia Ebinbodaus – Treeni ja Paino-ohjelmassa. Näiden ymmärtäminen on oleellista sovelluksen käyttön kannalta.

**TÄHÄN TERMEJÄ ESIM. TREENI = \_\_, SETTI=\_\_\_, SARJA=\_\_\_, PAINO=\_\_\_ jne.**

## Viitteet

Dokumentin sisällön sekä ohjelmistokehityksen tukena käytettiin seuraavia lähteitä.

Sulamo.fi

<https://www.sulamo.fi/>

ScottPlot.Net

<https://scottplot.net/>

Microsoft Learn

[https://learn.microsoft.com/](https://learn.microsoft.com/fi-fi/)

(**MAHDOLLISESTI LISÄTTÄVÄÄ?)**

## Yleiskatsaus dokumenttiin

Tässä dokumentissa käydään läpi Ebinbodaus – Treeni ja Paino-ohjelman toiminnalliset ja ei-toiminnalliset vaatimukset yksityiskohtaisesti ohjelman käyttäjille ja kehittäjille.

Dokumentti tarjoaa yksityiskohtaisen kuvauksen sovelluksen käyttöympäristöstä, toiminnasta, tietojen ja tietokannan käsittelystä, sovelluksen toiminnoista, ulkoisista liittymistä sekä muista ominaisuuksista. Lisäksi siinä käsitellään suunnittelurajoitteet, hylättyjä ratkaisuvaihtoehtoja ja esitetään jatkokehitysideoita.

# Yleiskatsaus

Tässä kappaleessa kuvataan Ebinbodaus – Treeni ja Paino -ohjelman käyttöympäristöä, toimintaa, käyttäjäkuntaa sekä yleisiä rajoituksia ja riippuvuuksia.



## Ympäristö

Ebinbodaus – Treeni ja Paino-ohjelma on tarkoitettu käytettäväksi kaikkialla missä oma paino on mahdollista mitata, kuten kuntosalilla tai kotona suorituksen jälkeen.

## Toiminta

Ohjelma kerää käyttäjän painon halutulta päivältä ja asettaa sen viivadiagrammiin visualisoimaan painonkehitystä viikon, kuukauden tai vuoden ajalta. Viivadiagrammin toteutukseen käytetään NuGet-ohjelmistopaketti ScottPlottia.

Käyttäjä voi myös kirjata treenimuistiota viikonajalta.

Ohjelman kokonaisuus toteutetaan Windows Presentation Foundation (WPF) .NET-kehyksellä, C# -ohjelmointikielellä ja kaikki data tallennetaan JSON-tiedostoksi.

## Käyttäjät

Ebinbodaus – Treeni ja Paino-ohjelma on suunnattu kaikille painon seurauksesta kiinnostuneille ja salitreeniä harrastaville.

## Yleiset rajoitteet

Ohjelman käyttöön tarvitaan tietokone, joka kykenee suorittamaan WPF sovellusta. Microsoftin .NET-framework tai .NET Core on oltava asennettuna tietokoneelle.

Ohjelma on ilmainen käyttää.

## Odotukset ja riippuvuudet

Käyttäjältä vaaditaan vaaka, jotta painon kirjaus on ohjelmassa mahdollista. Käyttäjältä odotetaan myös perusohjelmiston käyttökykyä, kuten tekstikenttiin kirjoittamista. Ohjelma on suunniteltu helppokäyttöiseksi ja käyttäjälle kerrotaan, jos tekemä syöte on mahdoton, virheitten estämiseksi.

# Tiedot ja tietokanta

Tässä kappaleessa kuvataan Ebinbodaus – Treeni ja Paino -ohjelman tiedonhallintaa, käyttöintensiteettiä sekä kapasiteettivaatimuksia.



## Tietosisältö

Tieto tallennetaan ohjelman suorituksen aikana kahdessa pääluokassa: *Painoni* *ja PaivaLista*.

Suorituksen aikana ohjelma luo uusia Painoni- ja PaivaLista-olioita ja lisää ne PainoLista- ja PaivaList -listoihin. Näitä listoja käytetään sitten datan säilyttämiseen ohjelman suorituksen aikana.

## Käyttöintensiteetti

Käyttäjää oletetaan merkitsemään painonsa ohjelmaan aina punnitsemisen jälkeen ja viikon treenimuistiota tulee seurata ja päivittää vähintään kaksi kertaa viikossa.

## Kapasiteettivaatimukset

Ohjelma voi käsitellä melko laajoja määriä dataa, mutta sen tarkka rajoitus riippuu useista tekijöistä, kuten käytetystä laitteistosta ja muista resursseista.

Yleisesti ottaen ScottPlotin avulla voi piirtää ja visualisoida suhteellisen suuria datamääriä tehokkaasti, mutta ohjelman toimivuutta ei ole testattu esimerkiksi usean vuoden ajalta kerätystä datasta.

## Tiedostot ja asetustiedostot

Ohjelma tallentaa käyttäjän lisäämät syötteet automaattisesti JSON-tiedostoon.

# Toiminnot

**Tähän lyhyt kuvaus sovelluksen toiminnasta. Eli mitä se tekee. Kuvaaja myös tähän!**



## Pääsivu ja navigointi

Ohjelman etusivulla käyttäjä näkee diagrammista viimeisen seitsemän päivän painoseuranta **diagrammin**. Sivun vasemmassa laidassa on **hampurilaisvalikko** mistä käyttäjä pääsee käsiksi treenipäiväkirjaan sekä painonseuranta sivulle.

***Kuva here!***

Hampurilaisvalikko on aina näkyvillä sivusta riippumatta, navigaation helpottamisen takaamiseksi.

### Pääsivun toiminta step-by-step koodissa:

**Alustus**: Ohjelma alustaa *MainWindow*-luokan ja sen käyttöliittymän (*MainWindow.xaml*). Luokka sisältää staattisen metodin *GetInstance()*, jonka avulla pyritään varmistamaan, että on olemassa vain yksi *MainWindow*-instanssi kerrallaan.

**JSON-tiedoston lukeminen**: Ohjelma lukee tiedot *JSON*-tiedostosta *data.json*. Tiedot deserialisoidaan *Dictionary<string, double>* -muotoon, jossa avaimena on päivämäärä ja arvona paino.

**Viimeiset 7 päivää**: Ohjelma järjestää datan päivämäärän perusteella laskevaan järjestykseen, ottaa talteen viimeiset 7 päivää ja järjestää ne takaisin nousevaan järjestykseen. Näin saadaan viimeiset 7 päivää datasta.

**Viivadiagramm**i: Käytetään *ScottPlot*-kirjastoa piirtämään viivadiagrammi, joka näyttää painon kehityksen viimeisen 7 päivän aikana.

Käyttöliittymän päivitys: Käyttöliittymäelementtien (Tg\_Btn, tt\_koti, tt\_treeni, tt\_paino, img\_bg) näkyvyys ja läpinäkyvyys asetetaan riippuen käyttäjän toiminnasta (esim. valikon avaus).

**Navigointi**: Käyttäjä voi navigoida eri näkymiin (pääruutu, treenit, painonseuranta) klikkaamalla valikossa olevia painikkeita. *Paaruutu\_Click, CloseBtn\_Click, Treenit\_Click ja PainoOsio\_Click* -metodit hallinnoivat näiden painikkeiden toimintaa.

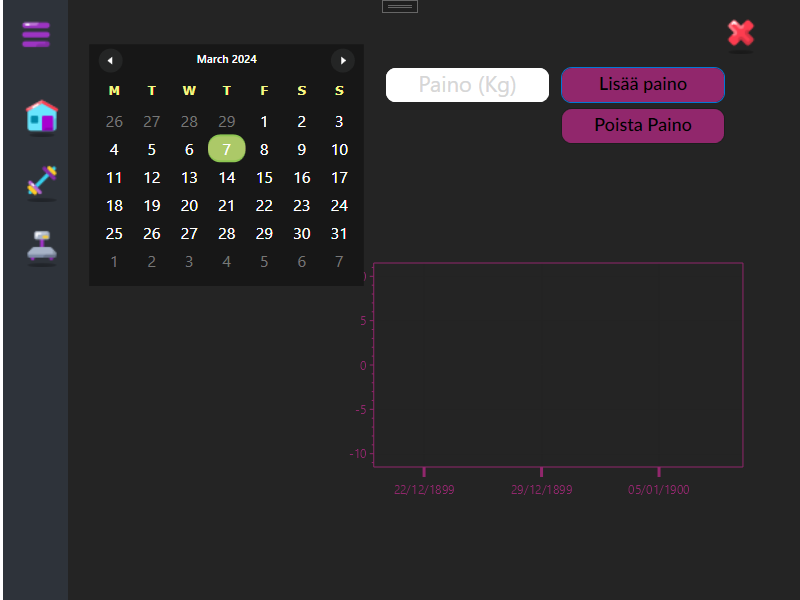
**Ikkunan siirtäminen**: Käyttäjä voi siirtää ikkunaa hiirellä vetämällä, mikä on toteutettu *Window\_MouseDown* -metodissa.

**Tooltippien hallint**a: Käyttöliittymässä on toteutettu toiminnallisuus, joka muuttaa tooltippien näkyvyyttä riippuen siitä, onko valikko avattu vai ei.

Ohjelma näyttää siis viimeisen 7 päivän painotiedot viivadiagrammissa ja tarjoaa käyttäjälle mahdollisuuden navigoida eri näkymien välillä. Lisäksi se sisältää toiminnallisuuksia käyttöliittymän ulkoasun muuttamiseen käyttäjän toimintojen perusteella.

## Painonseuranta

Ohjelman etusivulta painettua *Paino* -nappia, ohjelma avaa painonseuranta sivunäkymän. Ensimmäisellä käyttö kerralla sivulla näkyy **kalenteri**, **tekstikenttä**, **painon lisäys nappi**, **painon poisto nappi** sekä tyhjä **diagrammi**.

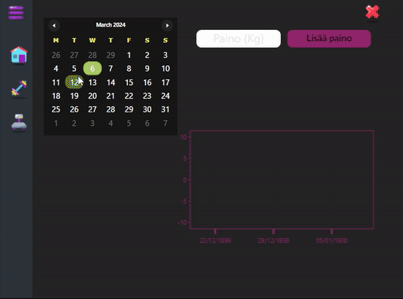


*Kuva -. Painonseuranta ensinäkymä*

Käyttäjä valitsee kalenterilta päivän, jolle hän haluaa merkitä painonsa. Kalenteri näyttää automaattisesti kyseisen päivän.

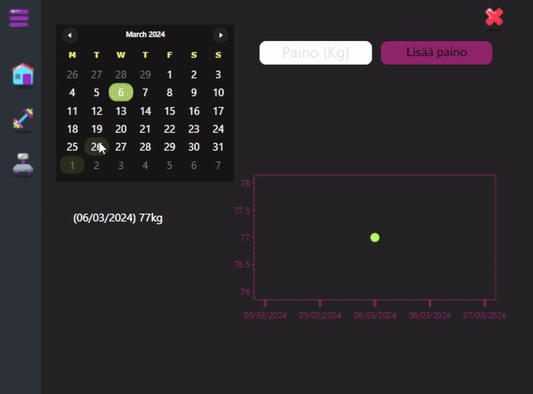
Tekstikenttään kirjoitetaan kirjattava paino *kilogrammoissa*, jonka jälkeen painetaan Lisää paino -nappia.

Kirjatun painon arvo tulee pisteenä näkyviin diagrammiin merkityn päivämäärän kohdalle. Sama arvo listataan myös listalle diagrammin viereen.



*Kuva -. Painon lisäys*

Toisen kirjauksen jälkeen diagrammi piirtää datan mukaisen viivan diagrammiin. Listattu dataa näkyy päiväjärjestyksessä diagrammin viereisellä listalla. Jos päivä on jäänyt välistä, käyttäjä voi huoletta valita sen ja lisätä kirjauksen, diagrammi päivittää listausten ja diagrammin pisteiden järjestyksen automaattisesti.



(Diagrammin päivitys)

Jokaiselle päivälle voi kirjata painon vain kerran ja tekstikenttään voi lisätä vain numeroita.

Jos kirjauksen haluaa poistaa, tulee valita haluttu päivämäärä ja painaa poista paino -nappia.



### Painonseurannan toiminta step-by-step koodissa:

**Alustus:** Ohjelma alustaa kaksi listatyyppistä muuttujaa *PainoLista* ja *PaivaList*, joissa säilytetään paino- ja päivämäärätietoja.

**Käyttöliittymän alustus:** *Painonseuranta()* -metodi alustaa käyttöliittymän asettamalla oletuspäivämäärän kalenterille ja määrittämällä alueen kaavion piirtämistä varten. Se myös lataa tiedot *JSON*-tiedostosta ja piirtää kaavion kutsuen *LoadDataFromJson()* ja *PlotData()* -metodeja.

**LoadDataFromJson():** Tämä metodi lukee tiedot *JSON*-tiedostosta, jos sellainen löytyy. Se muuntaa *JSON*-muotoisen datan takaisin *dates* ja *weights* -taulukoiksi, jotka edustavat päivämääriä ja painoja. Sen jälkeen se tyhjentää *PaivaList*- ja *PainoLista*-muuttujat ja täyttää ne *JSON*-tiedoston sisällöllä. Lopuksi se kutsuu *PaivitaPaino()* -metodia päivittääkseen käyttöliittymän tiedot.

**PlotData():** Tämä metodi lajittelee *PaivaList*- ja *PainoLista*-muuttujien sisällön päivämäärän mukaan, jotta kaavioon piirrettävät tiedot ovat oikeassa järjestyksessä. Sen jälkeen se piirtää kaavion, joka näyttää painojen kehityksen ajan suhteen.

**Button\_Click():** Tämä metodi käsittelee painikkeen napsautusta. Se tarkistaa, onko käyttäjän syöttämä paino kelvollinen ja onko valitulle päivämäärälle jo olemassa merkintä. Jos kaikki on kunnossa, se lisää uuden päivämäärän ja painon *PaivaList*- ja *PainoLista*-muuttujiin ja päivittää *JSON*-tiedoston. Sitten se piirtää kaavion uudelleen ja päivittää käyttöliittymän tiedot kutsuen *PaivitaPaino()* -metodia.

**SaveDataToJson():** Tämä metodi tallentaa päivämäärä- ja painotiedot *JSON*-tiedostoon.

**PaivitaPaino():** Tämä metodi päivittää käyttöliittymän näyttämään käyttäjälle kaikki tallennetut päivämäärät ja niihin liittyvät painot.

Ohjelman toiminta perustuu siis käyttäjän syöttämien painojen ja päivämäärien tallentamiseen listoihin, niiden näyttämiseen käyttöliittymässä ja niiden visualisointiin kaaviona. Lisäksi tiedot tallennetaan ja luetaan *JSON*-tiedostosta, jotta ne säilyvät käyttöjen välillä.

## Treenipäiväkirja

**Kuinka sovellus toimii? Tässä kohtaa olisi hyvä olla diagram/kuva sovellusken toiminnoista. Ns. Suunnittelu osuus. Voisi pitää mielessä miten sovellusta voisi mahdollisesti kehittää, mutta vain lyhyesti kuvassa.**

**Tässä olisi hyvä myös olla kuva myös itse sovelluksesta kun sen on tehnyt**

# Ulkoiset liittymät

Tässä kappaleessa kerrotaan Ebinbodaus – Treeni ja Paino -ohjelmanssa käytetyistä ulkoisista liittymistä.

## ScottPlot

Ebinbodaus – Treeni ja Paino -ohjelma hyödyntää datan viivadiagrammiksi visualisointiin kolmannen osapuolen avointa lähdekoodi kirjastoa ScottPlottia, joka tarjoaa graafisen kirjaston, joka on optimoitu tieteellisten kaavioiden piirtämiseen .NET frameworkissa.

Ohjelmaa luodessa on käytetty ScottPlotin sen hetkistä viimeisintä versioita ScottPlot 5.0.

# Muut ominaisuudet

Tässä kappaleessa määritellään Ebinbodaus – Treeni ja Paino -ohjelman muut ominaisuudet, kuten suorituskyky, turvallisuus, ylläpidettävyys ja käytettävyys.

## Suorituskyky ja vasteajat

Ohjelman tulee vastata käyttäjän pyyntöihin alle 2 sekunnin kuluessa. Suorituksien on myös oltava kevyitä tietokoneen muistille, myös datamäärien kasvaessa.

## Saavutettavuus (availability), toipuminen, turvallisuus, suojaukset

On varmistettava, että mahdollisten huolto/toimintakatkojen aiheuttamana asiakas ei menetä tallennettua dataansa.

Ohjelma on käyttäjän tietokoneelle ladattava työpöytä sovellus ja se tallentaa dataa asiakkaan koneelle. Asiakas pitää itse huolen ulkopuolisten tahojen pääsemisestä tietokoneelleen ja on täten itse vastuussa oman tietoturvallisuuden ylläpitämisestä.

## Ylläpidettävyys

Sovelluskehittäjät vastaavat sovelluksen päivittämisestä.

## Siirrettävyys ja yhteensopivuus

Tämä ohjelmisto rakennetaan käyttäen C#-kieltä Windows Presentation Foundation (WPF) -ympäristössä. UI-elementit kuvataan XAML:lla, joka on XML-pohjainen ohjelmointikieli. Sovellus suunnitellaan siten, että se on yhteensopiva tietokoneiden kanssa, joissa on käytössä Windows 7 tai uudempi käyttöjärjestelmä, mutta uudempi käyttöjärjestelmä on käyttökokemuksen kannalta suositeltu.

Ennen ohjelman julkaisemista on suositeltavaa suorittaa kattavat testit eri Windows-versioilla varmistaaksesi, että ohjelma toimii oikein erilaisissa ympäristöissä.

## Operointi

Käyttöliittymän tulisi olla helppokäyttöinen ja intuitiivinen ja järjestelmän tulee viestiä käyttäjälle virheilmoituksin epäonnistuneesta toiminnasta. Käytön aikana käyttäjä on itse vastuussa oman laitteensa tietoturvasta.

## Käytettävyys (Usability), käytön tehokkuus, käyttäjien tyytyväisyys

Käytön tehokkuuden sekä käyttäjien tyytyväisyyden takaamiseksi on tärkeää mahdollistaa käyttäjäpalaute tavalla tai toisella. Jatkokehitysajatuksissa käymme läpi mahdollisia tapoja joilla loppukäyttäjän ääni saadaan kuulolle.

# Suunnittelurajoitteet

Tässä kappaleessa käsitellään rajoitteita, jotka vaikuttavat ohjelman toteutukseen, hyödynnettyjä standardeja sekä muita mahdollisia rajoitteita.

## Standardit

Ebinbodaus – Treeni ja Paino -ohjelmassa hyödynnetään seuraavia standardeja:

**C#-kieli ja .NET Framework:** Ohjelma on kirjoitettu C#-kielellä ja käyttää .NET Frameworkia, mikä on Microsoftin kehittämä ohjelmistokehys Windows-sovellusten kehittämiseen.

**XAML (Extensible Application Markup Language):** XAML on XML-pohjainen merkintäkieli, jota käytetään WPF-sovellusten käyttöliittymien määrittämiseen.

**WPF (Windows Presentation Foundation):** WPF on Microsoftin kehittämä käyttöliittymätekniikka, jota käytetään Windows-sovellusten käyttöliittymien rakentamiseen.

**ScottPlot:** ScottPlot on grafiikkakirjasto, jota käytetään datan visualisointiin. Vaikka se ei ole standardi itsessään, se noudattaa yleisiä käytäntöjä ja periaatteita datan visualisoinnissa.

**Newtonsoft.Json:** Tämä on JSON-tiedostojen käsittelyyn tarkoitettu kirjasto. JSON on yleinen tiedostomuoto datan tallentamiseen ja välittämiseen.

## Laitteistorajoitteet

Ohjelma on WPF-sovellus, joten seuraavat tyypilliset laitteistorajoitteet on otettava huomioon kehityksessä.

**Suoritinresurssit:** WPF tekniikkaa käyttävät sovellukset voivat mahdollisesti vaatia tiettyä prosessori- sekä näytönohjaintehoa toimiakseen mahdollisimman nopeasti

**Näytön tarkkuus:** WPF:n käyttöliittymäelementit skaalautuvat saumattomasti eri näyttöresoluutioille hyödyntäen vektorigrafiikkaa. Tämä kuitenkin edellyttää tiettyä näytön tarkkuutta, jotta elementit näkyvät selkeästi ja ovat helposti käytettäviä.

**Muisti:** WPF-sovellukset voivat olla vaativia muistin määrästä, joten riittävä muisti on oleellinen vakaan kokemuksen takaamiseksi käytön aikana.

**Tallennustila:** Toimiakseen sovelluksella on oltava riittävästi tallennustilaa, varsinkin tallennetun datamäärän kasvaessa.

## Ohjelmistorajoitteet

Ohjelma on WPF-sovellus, joten seuraavat tyypilliset ohjelmistorajoitteet on otettava huomioon kehityksessä.

**Käyttöjärjestelmävaatimukset:** Ebinbodaus – Treeni ja Paino -ohjelma vaatii toimiakseen Windows 7 tai uudemman käyttöjärjestelmän.

**Käyttöliittymän skaalattavuus:** Ebinbodaus – Treeni ja Paino -ohjelma on suunniteltu tietylle resoluutiolle, eikä skaalaaminen ole mahdollista.

**.NET -versio:** Ebinbodaus – Treeni ja Paino -ohjelman kohde versio on .NET 8.0

**Kehitysympäristö:** Sovelluksen kehittämistä varten vaaditaan WPF käyttöliittymätekniikkaa hyödyntävä kehitysympäristö, kuten Visual Studio.

## Muut rajoitteet

Ebinbodaus – Treeni ja Paino -ohjelma on kehitetty noin neljän viikon kestäneessä aikaikkunassa hyödyntäen Scrum-työskentelymallia, jonka seurauksena kehitystä on priorisoitu vaadittavin tavoin aikataulussa pysymisen varmistamiseksi. Vaikka päätoimivuudet ovat kutakuinkin suunnitellusti kehitetty, jotkut parannukset sekä kehityksen aikana syntyneet ideat tullaan siirtämään jatkokehitykseen.

Sovelluksen kehityksessä hyödynnetty scrum on ketterä työskentelymalli, joka auttaa tiimejä kehittämään ja toimittamaan tuotteita tehokkaasti ja joustavasti. Lyhyesti sanottuna Scrum jakaa projektin lyhyisiin kehitysjaksoihin, joita kutsutaan sprinteiksi.

Jokaisessa sprintissä tiimi valitsee tietyn määrän työtehtäviä, jotka he aikovat saada valmiiksi sprintin aikana. Sprintin lopussa tiimi esittelee tehdyt työt ja kerää palautetta työskentelystä Review kokouksessa, jonka jälkeen (yleensä Review kokouksen yhteydessä) tiimi arvioi sprintin onnistumisen ja tunnistaa parannusmahdollisuudet tulevia sprinttejä varten Retrospective kokouksessa.

# Hylätyt ratkaisuvaihtoehdot

Ebinbodaus – Treeni ja Paino -ohjelman kehityksen aikana seuraavat ratkaisuvaihtoehdot hylättiin.

**WinForm: En nyt tässä väsymystilassa muista mikä siinä oli, lisää tähän**

**Lisätkää myös muita joita on ilmennyt.**

# Jatkokehitysajatuksia

Ideoita ja suunnitelmia siitä miten ohjelmistoa voisi mahdollisesti parantaa, kehittää tai laajentaa tulevaisuudessa.

# Liitteet

Tänne kaikki muut dokumentit ja tiedostot, jotka tukevat ja täydentävät vaatimusmäärittelyä. Näitä voivat olla mm. Taulukot, havainnollistavat kuvat, laskelmat tai muu vastaava.